Verbio Software Reference

Referencia de las funciones del CT ADE SDK

Verbio Technologies, S.L.

Verbio Software Reference: Referencia de las funciones del CT ADE SDK

Verbio Technologies, S.L.

publicado Septiembre de 2011





Tabla de contenidos

Lista de tablas



Lista de ejemplos



Capítulo 1. Introducción

El objectivo de este documento es describir el conjunto de funciones que constituyen el `CT ADE SDK` proporcionado en *Verbio*. Este `SDK` está diseñado pensando en aquellos integradores que trabajan en entornos de programación CT ADE (Parity Software) sobre tarjetas `Intel Dialogic`.

En el Capítulo 2. *Descripción de las funciones* se describen todas las funciones disponibles en el `SDK`. Todos los parámetros de las funciones y los resultados devueltos son `strings` de texto. Todas las funciones del SDK son funciones síncronas.

En el Capítulo 4. *Código de ejemplo* se incluye un ejemplo de utilización de este `SDK`.



Capítulo 2. Descripción de las funciones

1. ivr_activatevcb

FUNCION `ivr_activatevcb ( chdev, vcbhandle )`

`chdev`	descriptor de un canal abierto de Dialogic
`vcbhandle`	descriptor de un vocabulario o gramática ABNF previamente cargado

RETORNA >=0 indicando el descriptor del vocabulario si el resultado es satisfactorio

-1 en caso de error

1.1. Descripción

La función **activatevcb()** activa el vocabulario previamente cargado en el canal especificado. El descriptor del vocabulario cargado se obtiene invocando a la función **ivr_loadvcb()** o **ivr_loadcd()**. Cuando no deba utilizarse en el siguente proceso de reconocimiento, debe ser desactivado mediante la función **ivr_deactivatevcb()**.

1.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
vcbhandle	descriptor de un vocabulario o gramática ABNF previamente cargado mediante la función **ivr_loadvcb()** o **ivr_loadcd()**.

1.3. Precauciones

En cualquier momento, pueden estar activos varios vocabularios y gramáticas. El descriptor utilizado en la función **ivr_activatevcb()** debe ser el valor retornado por la función **ivr_loadvcb()** (o **ivr_loadcd()**). La función **ivr_ivcb()** retorna el descriptor asociado al vocabulario que contiene la locución reconocida.

Si se establece el vocabulario o la gramática activa mediante la función **ivr_setvcb()** o **ivr_setcd()**, se descargarán automáticamente todos los vocabularios y gramáticas cargados hasta ese momento.

1.4. Errores

`EVX_NOMEM`	OUT OF MEMORY
`EVX_VCBFILE`	THE VOCABULARY FILE NAME IS NOT VALID. Check the vocabulary file name and path writing permission.
`EVX_INVWORD`	THE VOCABULARY TRANSCRIPTION WAS NOT FOUND
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

1.5. Elementos relacionados

ivr_setvcb, ivr_loadvcb, ivr_loadcd, ivr_deactivatevcb, ivr_unloadvcb, ivr_ivcb

1.6. Códigos de ejemplo

Consulte el ejemplo.

2. ivr_asr_init

FUNCION	`ivr_asr_init( configuration, defasrlng)`
	`configuration` — identificador de la configuración de reconocimiento por defecto
	`defasrlng` — identificador del idioma de reconocimiento por defecto
RETORNA	0 si el resultado es satisfactorio
	-1 en caso de error

2.1. Descripción

La función **ivr_asr_init()** inicializa la librería de reconocimiento del habla de `Verbio`. En nuevos desarrollos es recomendable utilizar esta función (conjuntamente con la función **ivr_tts_init** en caso de que también se requiera síntesis del habla) en lugar de la función **ivr_libinit**, puesto que permite indicar al servidor los parámetros de inicialización de una manera mucho más concreta.

2.2. Descripción de los parámetros

configuration

especifica la configuración por defecto con la que arrancará el servidor de reconocimiento. Por configuración de reconocimiento se entiende el idioma o idiomas soportados (en caso de configuraciones multilingües) por el servidor de reconocimiento. Cualquier proceso de reconocimiento invocado usará los modelos de reconocimiento asociados con la configuración activa en ese momento, por lo que únicamente podrá reconocerse en alguno de los idiomas contenidos en dicha configuración. Para más información, puede consultarse el documento `Verbio Software Reference: Guía del Usuario`.

`"es"`	Español castellano
`"es_ca"`	Español castellano + catalán
`"es_eu"`	Español castellano + euskera
`"es_ga"`	Español castellano + gallego
`"es_ca_eu_ga"`	Español castellano + catalán + euskera + gallego
`"es-ar"`	Español argentino
`"es-cl"`	Español chileno
`"es-co"`	Español colombiano
`"es-mx"`	Español mexicano
`"es-ve"`	Español venezolano
`"pt"`	Portugués
`"pt-br"`	Portugués brasileño
`"fr"`	Francés

"en-us"	Inglés

defasrlng — especifica el idioma por defecto con el que trabaja la configuración por defecto. En el caso de configuraciones monolingües es redundante, aunque sí es imprescindible cuando se activa una configuración multilingüe.

"es"	Español castellano
"ca"	Catalán
"eu"	Euskera
"ga"	Gallego
"es-ar"	Español argentino
"es-cl"	Español chileno
"es-co"	Español colombiano
"es-mx"	Español mexicano
"es-ve"	Español venezolano
"pt"	Portugués
"pt-br"	Portugués brasileño
"fr"	Francés
"en-us"	Inglés

2.3. Precauciones

La función **ivr_asr_init()** debe invocarse antes que ninguna otra de las funciones del SDK de Verbio relacionadas con el reconocimiento del habla, a excepción de aquella destinada a establecer el servidor de trabajo por defecto en configuraciones cliente-servidor (**ivr_setparm**) o bien a su equivalente en síntesis (**ivr_tts_init**).

2.4. Errores

EVX_SRERROR	DIALOGIC SRL ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
EVX_DXERROR	DIALOGIC VOICE ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
EVX_NOBOARDS	NO BOARDS DETECTED. Check Dialogic Drivers Start Up.
EVX_INVSETUP	VERBIO INTERNAL ERROR. Files may be corrupted. Check disk and repeat Setup.
EVX_NOMEM	OUT OF MEMORY.
EVX_NOLICFILE	NO LICENSE FILE WAS FOUND. Run CheckOut to obtain a license file.
EVX_INVLIC	THE LICENSE FILE IS NOT VALID. Run CheckOut to obtain a valid license file.
EVX_SYSTEM	SYSTEM ERROR.Check errno.

`EVX_INVSRVER`	INVALID DIALOGIC SRL RELEASE (libsrlmt.dll). Check README.TXT.
`EVX_INVDXVER`	INVALID DIALOGIC VOICE LIBRARY RELEASE (libdxxmt.dll).

2.5. Elementos relacionados

ivr_libinit, ivr_tts_init

2.6. Códigos de ejemplo

Consulte el ejemplo.

3. ivr_builtin

FUNCION `ivr_builtin( chdev)`

`chdev` descriptor de un canal abierto de Dialogic

RETORNA 0 para indicar que el resultado del reconocimiento no procede de una gramática `built-in`

1 para indicar que el resultado del reconocimiento sí procede de una gramática `built-in`

3.1. Descripción

La función **ivr_builtin()** indica si la gramática que ha devuelto el resultado de reconocimiento al finalizar la llamada a la función **ivr_recind()** o **ivr_recresult()** o **ivr_recscore()** es una gramática interna (`built-in`) o no.

3.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

3.3. Precauciones

El valor devuelto por la función **ivr_builtin()** únicamente es válido después de la correcta finalización de la función **ivr_recind()** o **ivr_recresult()** o **ivr_recscore()**.

3.4. Elementos relacionados

ivr_recind, ivr_recresult, ivr_recscore

3.5. Códigos de ejemplo

Consulte el ejemplo.

4. ivr_deactivatevcb

FUNCION	`ivr_deactivatevcb( chdev, vcbhandle)`	
	`chdev`	descriptor de un canal abierto de Dialogic
	`vcbhandle`	descriptor de un vocabulario o gramática ABNF previamente cargado y activado
RETORNA	>=0 indicando el descriptor del vocabulario si el resultado es satisfactorio	
	-1 en caso de error	

4.1. Descripción

La función **ivr_deactivatevcb()** desactiva un vocabulario o gramática ABNF previamente cargado y activado en el canal especificado. El descriptor del vocabulario se obtiene al cargarlo mediante la función **ivr_loadvcb()** (o **ivr_loadcd()**) y también debe usarse para activarlo mediante la función **ivr_activatevcb()** o para descargarlo mediante la función **ivr_unloadvcb()**.

4.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
vcbhandle	descriptor de un vocabulario o gramática ABNF previamente cargado mediante la función **ivr_loadvcb()** (o **ivr_loadcd()**) y activado mediante la función **ivr_activatevcb()**.

4.3. Precauciones

En cualquier momento, pueden estar activos varios vocabularios y gramáticas. El descriptor utilizado en la función **ivr_deactivatevcb()** debe ser el valor retornado por la función **ivr_loadvcb()** (o **ivr_loadcd()**).

Si se establece el vocabulario o la gramática activa mediante la función **ivr_setvcb()** o **ivr_setcd()**, se descargarán automáticamente todos los vocabularios y gramáticas cargados hasta ese momento.

4.4. Errores

`EVX_NOMEM`	OUT OF MEMORY
`EVX_VCBFILE`	THE VOCABULARY FILE NAME IS NOT VALID.
`EVX_INVWORD`	THE VOCABULARY TRANSCRIPTION WAS OT FOUND
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

4.5. Elementos relacionados

ivr_setvcb, ivr_loadvcb, ivr_loadcd, ivr_activatevcb, ivr_unloadvcb, ivr_ivcb

4.6. Códigos de ejemplo

Consulte el ejemplo.

5. ivr_errorline

FUNCION `ivr_errorline()`

RETORNA número de la línea del fichero de vocabulario o gramática `ABNF` que contiene la palabra incorrecta

5.1. Descripción

La función **ivr_errorline()** retorna la línea que contiene la palabra incorrecta en un fichero de vocabulario o gramática `ABNF`. Esta función debe utilizarse en caso de error en la función **ivr_prevcb()** o **ivr_prebnf()**.

El indicador de la palabra coincide con la línea del fichero en la que se encuentra en los ficheros de vocabularios convencionales (una palabra del vocabulario por línea). No obstante, en las gramáticas `ABNF` esta regla no se cumple, por lo que no es precisa en el diagnóstico.

5.2. Elementos relacionados

ivr_prevcb, ivr_prebnf

5.3. Códigos de ejemplo

Consulte el ejemplo.

6. ivr_getasrlic

FUNCION `ivr_getasrlic( configuration)`

`configuration` identificador de la configuración de reconocimiento de la que se desean consultar las licencias disponibles

RETORNA Número de licencias máximas disponibles (adquiridas)

6.1. Descripción

La función **ivr_getasrlic()** consulta el número de licencias de la configuración especificada disponibles en total (adquiridas) en el servidor utilizado (aquél especificado antes de invocar a la función **ivr_libinit** o **ivr_asr_init**).

6.2. Descripción de los parámetros

configuration

especifica la configuración de reconocimiento de la que se desea conocer el número de licencias disponibles. Para más información, puede consultarse el documento `Verbio Software Reference: Guía del Usuario`.

"es"	Español castellano
"es_ca"	Español castellano + catalán
"es_eu"	Español castellano + euskera
"es_ga"	Español castellano + gallego
"es_ca_eu_ga"	Español castellano + catalán + euskera + gallego
"es-ar"	Español argentino
"es-cl"	Español chileno
"es-co"	Español colombiano
"es-mx"	Español mexicano
"es-ve"	Español venezolano
"pt"	Portugués
"pt-br"	Portugués brasileño
"fr"	Francés
"en-us"	Inglés

6.3. Precauciones

La función **ivr_getasrlic()** debe invocarse una vez establecida la comunicación con el servidor de reconocimiento al que se le solicita el número de licencias disponibles, entendiendo por disponibles el número máximo de peticiones de reconocimiento simultáneas que pueden realizarse, en ese servidor, para la

configuración especificada.

6.4. Elementos relacionados

ivr_getusedasrlic

ivr_getttslic

ivr_getusedttslic

ivr_libinit

ivr_asr_init

6.5. Códigos de ejemplo

Consulte el ejemplo.

7. ivr_getdtmf

FUNCION `ivr_getdtmf(chdev)`

`chdev` descriptor de un canal abierto de Dialogic

RETORNA tonos DTMF que han causado la detención de la reproducción sintética anterior

-1 en caso de error

7.1. Descripción

La función **ivr_getdtmf()** retorna el tono o tonos DTMF que han causado la interrupción de la síntesis anterior. Los tonos que pueden interrumpirla (y, por lo tanto, ser devueltos por esta función) son los que se han especificado mediante la función **ivr_setdtmf()**. En caso de que no se especifiquen mediante la función **ivr_setdtmf()**, la reproducción se interrumpirá ante cualquier tono, pero la función **ivr_getdtmf()** no podrá obtener el tono que ha causado la interrupción.

7.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

7.3. Elementos relacionados

ivr_setdtmf

7.4. Códigos de ejemplo

Consulte el ejemplo.

8. ivr_getparm

FUNCION `ivr_getparm( chdev, parm)`

`chdev`	descriptor de un canal abierto de Dialogic o -1 para parámetros globales de *VERBIO* (independientes del canal)
`parm`	parámetro del que se desea obtener el valor

RETORNA Valor del parámetro solicitado si el resultado es satisfactorio

Vacío en caso de error

8.1. Descripción

La función **ivr_getparm()** obtiene los valores actuales de los parámetros globales o dependientes del canal de *VERBIO*. Esta función puede obtener el valor de un único parámetro en cada invocación.

8.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic. Use `-1` para obtener el valor de parámetros globales de *VERBIO*.

parm especifica el identificador del parámetro del que se desea obtener el valor. Para conocer los identificadores válidos, consulte el Capítulo 3. *Estructuras de datos y parámetros*.

8.3. Elementos relacionados

ivr_setparm

8.4. Códigos de ejemplo

Consulte el ejemplo.

9. ivr_getttslic

FUNCION `ivr_getttslic( language)`

`language` identificador del idioma de síntesis del que se desean consultar las licencias disponibles

RETORNA Número de licencias máximas disponibles (adquiridas)

9.1. Descripción

La función **ivr_getttslic()** consulta el número de licencias del idioma especificado disponibles en total (adquiridas) en el servidor utilizado (aquél especificado antes de invocar a la función **ivr_libinit** o **ivr_tts_init**).

9.2. Descripción de los parámetros

language especifica el idioma de síntesis del que se desea conocer el número de licencias disponibles. Para más información, puede consultarse el documento `Verbio Software Reference: Guía del Usuario`.

`"es"`	Español castellano
`"ca"`	Catalán
`"eu"`	Euskera
`"ga"`	Gallego
`"es-mx"`	Español mexicano
`"pt"`	Portugués
`"pt-br"`	Portugués brasileño*
`"fr"`	Francés*
`"en-us"`	Inglés*

(*) consultar condiciones particulares

9.3. Precauciones

La función **ivr_getttslic()** debe invocarse una vez establecida la comunicación con el servidor de síntesis al que se le solicita el número de licencias disponibles, entendiendo por disponibles el número máximo de peticiones de síntesis simultáneas que pueden realizarse, en ese servidor, para el idioma especificado.

9.4. Elementos relacionados

ivr_getusedttslic

ivr_getasrlic

ivr_getusedasrlic

ivr_libinit

ivr_tts_init

9.5. Códigos de ejemplo

Consulte el ejemplo.

10. ivr_getusedasrlic

FUNCION int `ivr_getusedasrlic()`

RETORNA Número de licencias disponibles en el momento de la invocación

10.1. Descripción

La función **ivr_getusedasrlic()** consulta el número de licencias de la configuración especificada disponibles en un momento determinado (libres) en el servidor utilizado (aquél especificado antes de invocar a la función **ivr_libinit** o **ivr_asr_init**).

10.2. Precauciones

La función **ivr_getusedasrlic()** devuelve el número de licencias disponibles en un momento determinado, para controlar la disponibilidad de las mismas antes de realizar una petición. Puede combinarse también con las funciones **ivr_recstr_wait** y **ivr_recstr_release**.

10.3. Elementos relacionados

ivr_getttslic, ivr_recstr_wait, ivr_recstr_release

10.4. Códigos de ejemplo

Consulte el ejemplo.

11. ivr_getusedttslic

FUNCION int `ivr_getusedttslic`()

RETORNA Número de licencias disponibles en el momento de la invocación

11.1. Descripción

La función **ivr_getusedttslic()** consulta el número de licencias de la configuración especificada disponibles en un momento determinado (libres) en el servidor utilizado (aquél especificado antes de invocar a la función **ivr_libinit** o **ivr_tts_init**).

11.2. Precauciones

La función **ivr_getusedttslic()** devuelve el número de licencias disponibles en un momento determinado, para controlar la disponibilidad de las mismas antes de realizar una petición. Puede combinarse también con las funciones **ivr_playstr_wait** y **ivr_playstr_release**.

11.3. Elementos relacionados

ivr_getusedasrlic, ivr_playstr_wait, ivr_playstr_release

11.4. Códigos de ejemplo

Consulte el ejemplo.

12. ivr_id

FUNCION `ivr_id( chdev, index)`

`chdev`	descriptor de un canal abierto de Dialogic
`index`	número de la palabra devuelta por el reconocedor de la que se desea conocer más información

RETORNA índice de la palabra reconocida si el resultado es satisfactorio

-1000 si no se ha reconocido ninguna palabra

-1001 si el usuario ha empezado a hablar antes de que empezara el reconocimiento

-1002 si no se ha detectado voz

-1 en caso de error

12.1. Descripción

La función **ivr_id()** devuelve el índice de la palabra reconocida en la posición especificada como parámetro. Este índice debe pasarse a la función **ivr_word** para obtener el resultado del reconocimiento. En caso de que el valor sea negativo, puede indicar alguna condición de terminación anormal, tal como se ha indicado en la sección anterior.

12.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
index	Número de la palabra devuelta por el reconocedor de la que se quiere obtener el resultado. El valor de este parámetro debe estar comprendido entre 0 y el número de palabras devuelto por el reconocedor, indicado como resultado de la función **ivr_recind()**.

12.3. Elementos relacionados

ivr_recind, ivr_word

12.4. Códigos de ejemplo

Consulte el ejemplo.

13. ivr_ivcb

FUNCION `ivr_ivcb(` `chdev`)

`chdev` descriptor de un canal abierto de Dialogic

RETORNA >=0 indicando el descriptor del vocabulario afectado por el último resultado de reconocimiento

-1 en caso de error

13.1. Descripción

La función **ivr_ivcb()** retorna el descriptor del vocabulario del que se ha reconocido una secuencia durante el último proceso de reconocimiento. Este valor está disponible después de la finalización de una llamada a las funciones **ivr_recind()** o **ivr_recresult()** o **ivr_recscore()**.

13.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

13.3. Precauciones

El valor devuelto por la función **ivr_ivcb** sólo es válido tras la correcta finalización de la función **ivr_recind()** o **ivr_recresult()** o **ivr_recscore()**.

13.4. Errores

`EVX_NORECIND` NO RECIND. Call **ivr_recind()** before calling **ivr_ivcb**.

13.5. Elementos relacionados

ivr_recind, ivr_recresult, ivr_recscore

13.6. Códigos de ejemplo

Consulte el ejemplo.

14. ivr_libinit

FUNCION	`ivr_libinit( options)`	
	`options`	identificador del idioma por defecto
RETORNA	0 si el resultado es satisfactorio	
	-1 en caso de error	

14.1. Descripción

La función **ivr_libinit()** inicializa la librería `Verbio`. El parámetro `options` especifica el idioma por defecto que utilizarán el servidor de reconocimiento y el servidor de síntesis.

Esta función está incluida por compatibilidad con versiones anteriores, aunque en nuevos desarrollos es muy recomendable utilizar las nuevas funciones equivalentes **ivr_asr_init** y/o **ivr_tts_init**, según las necesidades concretas.

14.2. Descripción de los parámetros

options — especifica el idioma de trabajo por defecto y si deben inicializarse los servidores de reconocimiento y de síntesis (por defecto, ambos se inicializan).

`"0x00000000"`	Español
`"0x01000000"`	Catalán
`"0x04000000"`	Euskera
`"0x02000000"`	Portugués
`"0x08000000"`	Portugués brasileño
`"0x00040000"`	no se inicializa el servidor de reconocimiento
`"0x00020000"`	no se inicializa el servidor de síntesis
`"0x00080000"`	inicialización rápida (menores prestaciones) del sintetizador (depuración)

14.3. Precauciones

La función **ivr_libinit()** debe invocarse antes que ninguna otra de las funciones del SDK de `Verbio`, a excepción de aquella destinada a establecer el servidor de trabajo por defecto en configuraciones cliente-servidor (**ivr_setparm**).

14.4. Errores

`EVX_SRERROR`	DIALOGIC SRL ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_DXERROR`	DIALOGIC VOICE ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_NOBOARDS`	NO BOARDS DETECTED. Check Dialogic Drivers Start Up.

`EVX_INVSETUP`	VERBIO INTERNAL ERROR. Files may be corrupted. Check disk and repeat Setup.
`EVX_NOMEM`	OUT OF MEMORY.
`EVX_NOLICFILE`	NO LICENSE FILE WAS FOUND. Run CheckOut to obtain a license file.
`EVX_INVLIC`	THE LICENSE FILE IS NOT VALID. Run CheckOut to obtain a valid license file.
`EVX_SYSTEM`	SYSTEM ERROR.Check errno.
`EVX_INVSRVER`	INVALID DIALOGIC SRL RELEASE (libsrlmt.dll). Check README.TXT.
`EVX_INVDXVER`	INVALID DIALOGIC VOICE LIBRARY RELEASE (libdxxmt.dll).

14.5. Elementos relacionados

ivr_asr_init

ivr_tts_init

14.6. Códigos de ejemplo

Consulte el ejemplo.

15. ivr_loadcd

FUNCION	`ivr_loadcd( chdev)`	
	`chdev`	descriptor de un canal abierto de Dialogic
RETORNA	>=0 indicando el descriptor del vocabulario si el resultado es satisfactorio	
	-1 en caso de error	

15.1. Descripción

La función **ivr_loadcd()** carga el vocabulario de dígitos conectados (consulte Sección 46 para más información) en el canal especificado. En caso satisfatorio, retorna un descriptor asociado con el vocabulario cargado para que éste pueda ser activado, desactivado y descargado mediante las funciones **ivr_activatevcb()**, **ivr_deactivatevcb()** y **ivr_unloadvcb()** respectivamente.

15.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

15.3. Precauciones

Para poder utilizar el vocabulario de dígitos conectados en un proceso de reconocimiento, es imprescindible activarlo previamente mediante la función **ivr_activatevcb**.

Si se establece el vocabulario o la gramática activa mediante la función **ivr_setvcb()** o **ivr_setcd()**, se descargarán automáticamente todos los vocabularios y gramáticas cargados hasta ese momento.

15.4. Errores

`EVX_NOMEM`	OUT OF MEMORY
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

15.5. Elementos relacionados

ivr_setcd, ivr_loadvcb, ivr_activatevcb, ivr_deactivatevcb, ivr_unloadvcb, ivr_ivcb

16. ivr_loadvcb

FUNCION `ivr_loadvcb(chdev, filename, mode)`

`chdev`	descriptor de un canal abierto de Dialogic
`filename`	nombre del vocabulario o de la gramática `ABNF` que debe cargarse
`mode`	vocabulario convencional o gramática `ABNF`

RETORNA >=0 indicando el descriptor del vocabulario si el resultado es satisfactorio

-1 en caso de error

16.1. Descripción

La función **ivr_loadvcb()** carga el vocabulario o la gramática `ABNF` en el canal especificado. En caso satisfatorio, retorna un descriptor asociado con el vocabulario cargado para que éste pueda ser activado, desactivado y descargado mediante las funciones **ivr_activatevcb()**, **ivr_deactivatevcb()** y **ivr_unloadvcb()** respectivamente.

16.2. Descripción de los parámetros

chdev — especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

filename — nombre del fichero de texto que contiene el vocabulario (cada línea contiene una `palabra del vocabulario` formada por una o más palabras) o la gramática `ABNF`.

mode — indica el tipo de vocabulario o gramática.

`"0x00010000"`	Palabras aisladas. El reconocedor espera reconocer una única `palabra del vocabulario` de este vocabulario.
`"0x00000000"`	Palabras conectadas (valor por defecto). El reconocedor espera reconocer una o más `palabras del vocabulario` de este vocabulario.
`"0x00080000"`	gramática `ABNF`. El reconocedor espera reconocer una o más palabras de la gramática.

16.3. Precauciones

Para poder utilizar el vocabulario cargado durante un proceso de reconocimiento, es necesario activarlo previamente mediante la función **ivr_activatevcb()**.

Sin la opción `"0x00080000"` habilitada, la función **ivr_loadvcb()** requiere el fichero de definiciones y el fichero de transcripciones generados mediante la función **ivr_prevcb()** (o **ivr_prevcbex()** o **ivr_prevcbdev()**). Si estos ficheros existen y el fichero de vocabulario correspondiente no ha sido modificado desde que se llamó a la función **ivr_prevcb()** (o **ivr_prevcbex()** o **ivr_prevcbdev()**), no es necesario volver a invocarlas antes de cargar el vocabulario mediante la función **ivr_loadvcb()**.

Con la opción `"0x00080000"` habilitada, la función **ivr_loadvcb()** requiere el fichero de estados de la gramática, el fichero de definiciones y el fichero de transcripciones generado mediante la función **ivr_prebnf()** (o **ivr_prebnfex()** o **ivr_prebnfdev()**). Si estos ficheros existen y la gramática `ABNF` correspondiente no ha sido modificado desde que se llamó a la función **ivr_prebnf()** (o **ivr_prebnfex()** o **ivr_prebnfdev()**), no es

necesario volver a invocarlas antes de cargar la gramática mediante la función **ivr_loadvcb()**.

Si se establece el vocabulario o la gramática activa mediante la función **ivr_setvcb()** o **ivr_setcd()**, se descargarán automáticamente todos los vocabularios y gramáticas cargados hasta ese momento.

16.4. Errores

`EVX_NOMEM`	OUT OF MEMORY.
`EVX_VCBFILE`	THE VOCABULARY FILE NAME IS NOT VALID.
`EVX_INVWORD`	THE VOCABULARY TRANSCRIPTION WAS NOT FOUND.
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

16.5. Elementos relacionados

ivr_setcd, ivr_loadcd, ivr_activatevcb, ivr_deactivatevcb, ivr_unloadvcb, ivr_ivcb

16.6. Códigos de ejemplo

Consulte el ejemplo.

17. ivr_nind

FUNCION `ivr_nind( chdev)`

`chdev` descriptor de un canal abierto de Dialogic

RETORNA >=0 para indicar el número de `palabras del vocabulario` o de la gramática reconocidas

-1 en caso de error

17.1. Descripción

La función **ivr_nind()** retorna el número de `palabras del vocabulario` o de la gramática reconocidas después de la finalización de la llamada a la función **ivr_recind()** o **ivr_recresult()** o **ivr_recscore()**.

17.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

17.3. Precauciones

El valor devuelto por la función **ivr_nind()** únicamente es válido después de la correcta finalización de la función **ivr_recind()** o **ivr_recresult()** o **ivr_recscore()**.

17.4. Errores

`EVX_NORECIND` NO RECIND. Call **ivr_recind()** before calling **ivr_nind()**.

17.5. Elementos relacionados

ivr_recind, ivr_recresult, ivr_recscore

17.6. Códigos de ejemplo

Consulte el ejemplo.

18. ivr_play2file

FUNCION	`ivr_play2file(chdev, string, filename)`	
	`chdev`	descriptor de un canal abierto de Dialogic
	`string`	texto que debe ser sintetizado
	`filename`	nombre del fichero en el que se almacenará el resultado de la síntesis
RETORNA	0 si el resultado es satisfactorio	
	-1 en caso de error	

18.1. Descripción

La función **ivr_play2file()** inicia la conversión de texto a voz del texto especificado y almacena las muestras resultantes en el fichero especificado.

18.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta telefónica.
string	texto que debe ser sintetizado. Para una óptima síntesis, es imprescindible que el texto esté bien puntuado, acentuado, etc.
filename	nombre del fichero en el que se almacenarán las muestras de voz sintéticas.

18.3. Elementos relacionados

ivr_playstr, ivr_playfile ivr_playrecstr, ivr_playrecfile, ivr_playrecaudio

18.4. Códigos de ejemplo

Consulte el ejemplo.

19. ivr_playfile

FUNCION	`ivr_playfile(chdev, filename)`	
	`chdev`	descriptor de un canal abierto de Dialogic
	`filename`	nombre del fichero que contiene el texto que debe ser sintetizado
RETORNA	0 si el resultado es satisfactorio	
	1 si se ha recibido un tono DTMF durante el proceso de síntesis del habla	
	-1 en caso de error	

19.1. Descripción

La función **ivr_playfile()** inicia la conversión de texto a voz del texto contenido en el fichero especificado. Una vez la función es invocada, la conversión continua hasta el final del texto, hasta que se detenga la actividad en el canal o hasta que se recibe un tono DTMF.

19.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta telefónica.
filename	nombre del fichero que contiene el texto que debe ser sintetizado. Para una óptima síntesis, es imprescindible que el texto esté bien puntuado, acentuado, etc.

19.3. Errores

`EVX_SRERROR`	DIALOGIC SRL ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_DXERROR`	DIALOGIC VOICE ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

19.4. Elementos relacionados

ivr_playstr, ivr_playrecstr, ivr_playrecfile, ivr_playrecaudio, ivr_playfilename

19.5. Códigos de ejemplo

Consulte el ejemplo.

20. ivr_playfilename

FUNCION	`ivr_playfilename(chdev, filename)`	
	`chdev`	descriptor de un canal abierto de Dialogic
	`filename`	nombre del fichero que contiene el audio que debe ser reproducido
RETORNA	0 si el resultado es satisfactorio	
	1 si se ha recibido un tono DTMF durante el proceso de reproducción	
	-1 en caso de error	

20.1. Descripción

La función **ivr_playfilename()** inicia la reproducción de las muestras de audio contenidas en el fichero especificado. El formato de las muestras debe ser VOX a 8 KHz, 4 bits/muestra ADPCM. Una vez la función es invocada, la conversión continua hasta la reproducción de todas las muestras, hasta que se detenga la actividad en el canal o hasta que se recibe un tono DTMF.

20.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta telefónica.
filename	nombre del fichero que contiene las muestras de audio a reproducir.

20.3. Errores

`EVX_SRERROR`	DIALOGIC SRL ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_DXERROR`	DIALOGIC VOICE ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

20.4. Elementos relacionados

ivr_playstr, ivr_playrecstr, ivr_playrecfile, ivr_playrecaudio, ivr_playfile

20.5. Códigos de ejemplo

Consulte el ejemplo.

21. ivr_playrecaudio

FUNCION `ivr_playrecaudio(chdev, filename, mode, timeout, maxsil)`

`chdev`	descriptor de un canal abierto de Dialogic
`filename`	nombre del fichero que contiene el fichero de audio a reproducir
`mode`	formato del fichero de audio (`A-law` o `Mu-law`)
`timeout`	tiempo máximo, en décimas de segundo, que puede durar el proceso de reconocimiento
`maxsil`	silencio máximo, en décimas de segundo, después de detectar voz

RETORNA 0 si el resultado es satisfactorio

1 si el reconocimiento ha finalizado por haber detectado un tono DTMF

7 si el reconocimiento ha finalizado por haberse completado el tiempo máximo (timeout)

-1 en caso de error

21.1. Descripción

La función **ivr_playrecaudio()** permite le ejecución simultánea de reproducción de audio y reconocimiento del habla en aquellos entornos que dispongan de una tarjeta con cancelación de eco. De este modo, puede interrumpirse el proceso de reproducción en cualquier momento, posibilitando unos diálogos más fluidos. Una vez detectada señal se detiene el proceso de reproducción y el proceso de reconocimiento continua hasta que se detengan las operaciones en el canal o se cumpla alguna de las condiciones de terminación: recepción de un tono DTMF o cumplimiento de alguna condición temporal especificada mediante los parámetros.

21.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

string nombre del fichero que contiene las muestras de audio a reproducir. En caso de que el usuario hable antes de que finalice, la reproducción se interrumpirá.

mode formato de las muestras de audio contenidas en el fichero especificado.

`"0x00000002"`	A-law
`"0x00000004"`	Mu-law

timeout tiempo máximo (expresado en décimas de segundo) que puede durar el proceso de reconocimiento. Transcurrido este tiempo, la función finaliza.

maxsil tiempo máximo (en décimas de segundo) de silencio que el reconocedor espera después de detectar voz. Una vez detectado este intervalo de silencio, la función finaliza.

21.3. Precauciones

La función **ivr_setvcb()** o **ivr_loadvcb()** debe ser llamada previamente a la función **ivr_playrecaudio()** para establecer el vocabulario o gramática activo.

Las tarjetas con cancelación de eco suelen tener que configurarse adecuadamente para poder utilizar la técnica

del `barge-in`. Consulte con el distribuidor para asegurarse de que el proceso se ha realizado correctamente.

21.4. Errores

`EVX_SRERROR`	DIALOGIC SRL ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_DXERROR`	DIALOGIC VOICE ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.
`EVX_NOSETVCB`	NO ACTIVE VOCABULARY. Use **ivr_setvcb()** to set the active vocabulary.

21.5. Elementos relacionados

ivr_playrecstr, ivr_playrecfile

21.6. Códigos de ejemplo

Consulte el ejemplo.

22. ivr_playrecfile

FUNCION `ivr_playrecfile`(`chdev`, `filename`, `mode`, `timeout`, `maxsil`)

`chdev`	descriptor de un canal abierto de Dialogic
`filename`	nombre del fichero que contiene el texto que debe ser sintetizado
`mode`	uso futuro
`timeout`	tiempo máximo, en décimas de segundo, que puede durar el proceso de reconocimiento
`maxsil`	silencio máximo, en décimas de segundo, después de detectar voz

RETORNA 0 si el resultado es satisfactorio

1 si el reconocimiento ha finalizado por haber detectado un tono DTMF

7 si el reconocimiento ha finalizado por haberse completado el tiempo máximo (timeout)

-1 en caso de error

22.1. Descripción

La función **ivr_playrecfile()** permite le ejecución simultánea de síntesis del habla y reconcimiento del habla en aquellos entornos que dispongan de una tarjeta con cancelación de eco. De este modo, puede interrumpirse el proceso de síntesis en cualquier momento, posibilitando unos diálogos más fluidos. Una vez detectada señal se detiene el proceso de síntesis y el proceso de reconocimiento continua hasta que se detengan las operaciones en el canal o se cumpla alguna de las condiciones de terminación: recepción de un tono DTMF o cumplimiento de alguna condición temporal especificada mediante los parámetros.

22.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
filename	nombre del fichero que contiene el texto que debe ser sintetizado. Para una óptima síntesis, es imprescindible que el texto esté bien puntuado, acentuado, etc. En caso de que el usuario hable antes de que finalice, la reproducción se interrumpirá.
mode	uso futuro (dejar a "0").
timeout	tiempo máximo (expresado en décimas de segundo) que puede durar el proceso de reconocimiento. Transcurrido este tiempo, la función finaliza.
maxsil	tiempo máximo (en décimas de segundo) de silencio que el reconocedor espera después de detectar voz. Una vez detectado este intervalo de silencio, la función finaliza.

22.3. Precauciones

La función **ivr_setvcb()** o **ivr_loadvcb()** debe ser llamada previamente a la función **ivr_playrecfile()** para establecer el vocabulario o gramática activo.

Las tarjetas con cancelación de eco suelen tener que configurarse adecuadamente para poder utilizar la técnica del `barge-in`. Consulte con el distribuidor para asegurarse de que el proceso se ha realizado correctamente.

22.4. Errores

`EVX_SRERROR`	DIALOGIC SRL ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_DXERROR`	DIALOGIC VOICE ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.
`EVX_NOSETVCB`	NO ACTIVE VOCABULARY. Use **ivr_setvcb()** to set the active vocabulary.

22.5. Elementos relacionados

ivr_playrecstr, ivr_playrecaudio

22.6. Códigos de ejemplo

Consulte el ejemplo.

23. ivr_playrecstr

FUNCION `ivr_playrecstr`(`chdev`, `string`, `mode`, `timeout`, `maxsil`)

`chdev`	descriptor de un canal abierto de Dialogic
`string`	texto que debe ser sintetizado
`mode`	uso futuro
`timeout`	tiempo máximo, en décimas de segundo, que puede durar el proceso de reconocimiento
`maxsil`	silencio máximo, en décimas de segundo, después de detectar voz

RETORNA 0 si el resultado es satisfactorio

1 si el reconocimiento ha finalizado por haber detectado un tono DTMF

7 si el reconocimiento ha finalizado por haberse completado el tiempo máximo (timeout)

-1 en caso de error

23.1. Descripción

La función **ivr_playrecstr()** permite le ejecución simultánea de síntesis del habla y reconcimiento del habla en aquellos entornos que dispongan de una tarjeta con cancelación de eco. De este modo, puede interrumpirse el proceso de síntesis en cualquier momento, posibilitando unos diálogos más fluidos. Una vez detectada señal se detiene el proceso de síntesis y el proceso de reconocimiento continua hasta que se detengan las operaciones en el canal o se cumpla alguna de las condiciones de terminación: recepción de un tono DTMF o cumplimiento de alguna condición temporal especificada mediante los parámetros.

23.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
string	texto que debe ser sintetizado. En caso de que el usuario hable antes de que finalice, la reproducción se interrumpirá.
mode	uso futuro (dejar a "0").
timeout	tiempo máximo (expresado en décimas de segundo) que puede durar el proceso de reconocimiento. Transcurrido este tiempo, la función finaliza.
maxsil	tiempo máximo (en décimas de segundo) de silencio que el reconocedor espera después de detectar voz. Una vez detectado este intervalo de silencio, la función finaliza.

23.3. Precauciones

La función **ivr_setvcb()** o **ivr_loadvcb()** debe ser llamada previamente a la función **ivr_playrecstr()** para establecer el vocabulario o gramática activo.

Las tarjetas con cancelación de eco suelen tener que configurarse adecuadamente para poder utilizar la técnica del `barge-in`. Consulte con el distribuidor para asegurarse de que el proceso se ha realizado correctamente.

23.4. Errores

`EVX_SRERROR`	DIALOGIC SRL ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_DXERROR`	DIALOGIC VOICE ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.
`EVX_NOSETVCB`	NO ACTIVE VOCABULARY. Use **ivr_setvcb()** to set the active vocabulary.

23.5. Elementos relacionados

ivr_playrecfile, ivr_playrecaudio

23.6. Códigos de ejemplo

Consulte el ejemplo.

24. ivr_playstr

FUNCION `ivr_playstr(chdev, string)`

`chdev`	descriptor de un canal abierto de Dialogic
`string`	texto que debe ser sintetizado

RETORNA 0 si el resultado es satisfactorio

1 si se ha recibido un tono DTMF durante el proceso de síntesis del habla

-1 en caso de error

24.1. Descripción

La función **ivr_playstr()** inicia la conversión de texto a voz del texto especificado. Una vez la función es invocada, la conversión continua hasta el final del texto, hasta que se detenga la actividad en el canal o hasta que se recibe un tono DTMF.

24.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta telefónica.
string	texto que debe ser sintetizado. Para una óptima síntesis, es imprescindible que el texto esté bien puntuado, acentuado, etc.

24.3. Errores

`EVX_SRERROR`	DIALOGIC SRL ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_DXERROR`	DIALOGIC VOICE ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

24.4. Elementos relacionados

ivr_playfile, ivr_playrecstr, ivr_playrecfile, ivr_playrecaudio

24.5. Códigos de ejemplo

Consulte el ejemplo.

25. ivr_playstr_release

FUNCION `ivr_playstr_release(chdev)`

`chdev`	descriptor de un canal abierto de Dialogic

RETORNA 0 si el resultado es satisfactorio

-1 en caso de error

25.1. Descripción

La función **ivr_playstr_release()** libera un recurso (licencia) de síntesis del habla reservado mediante la función **ivr_playstr_wait()** en el canal especificado.

25.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

25.3. Elementos relacionados

ivr_playstr_wait, ivr_getusedttslic, ivr_recstr_release, ivr_recstr_wait

25.4. Códigos de ejemplo

Consulte el ejemplo.

26. ivr_playstr_wait

FUNCION `ivr_playstr_wait(chdev, ms)`

`chdev`	descriptor de un canal abierto de Dialogic
`ms`	tiempo máximo de espera para obtener un recurso de síntesis (en milisegundos)

RETORNA 0 si el resultado es satisfactorio

-1 en caso de no poder obtener un recurso en el tiempo especificado

26.1. Descripción

La función **ivr_playstr_wait()** reserva un recurso (licencia) de síntesis del habla para el canal especificado hasta que se libere mediante la función **ivr_playstr_release()**. La función espera un máximo de `ms` milisegundos la disponibilidad de un recurso, transcurridos los cuales devuelve un error.

El uso de esta función es aconsejable para gestionar el uso de los recursos (licencias) disponibles. En entornos en los que nunca se producirán más peticiones simultáneas de síntesis que licencias disponibles no es necesario su uso.

26.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
ms	tiempo máximo de espera, en milisegundos, para obtener un recurso (licencia) de síntesis del habla para el canal especificado.

26.3. Elementos relacionados

ivr_playstr_release, ivr_getusedttslic, ivr_recstr_wait, ivr_recstr_release

26.4. Códigos de ejemplo

Consulte el ejemplo.

27. ivr_prebnf

FUNCION	`ivr_prebnf(filename)`	
	`filename`	nombre del fichero que contiene la gramática `ABNF` a preparar
RETORNA	0 si el resultado es satisfactorio	
	-1 en caso de error	

27.1. Descripción

La función **ivr_prebnf()** compila la gramática especificada. El resultado de la compilación se almacena en un nuevo fichero de texto con el mismo nombre que la gramática pero con extensión `".trx"` (conteniendo el vocabulario de la gramática) y en un nuevo fichero de texto con extensión `".grm"` (conteniendo los estados de la gramática). También se realiza la trancripción fonética que se almacena en un nuevo fichero binario con el mismo nombre que el de la gramática pero con extensión `".trc"`. Esta función es independiente del canal y debe utilizarse una vez para cada gramática aunque ésta se utilice en distintos canales (sí es necesario invocar de nuevo a la función cuando se modifica dinámicamente el contenido de la gramática). En caso que deba preparar gramáticas dinámicas en distintos canales, debe utilizarse la función **ivr_prebnfdev** para evitar conflictos entre ellos.

La gramática `ABNF` se compila siguiendo las reglas fonéticas del idioma establecido por defecto al inicializar el motor de reconocimiento, al invocar a la función **ivr_libinit** o **ivr_asr_init** o bien al modificar el parámetro `"0x80000403"` mediante la función **ivr_setparm**. En caso de querer preparar la gramática `ABNF` en un idioma conocido independiente del idioma por defecto, puede utililzarse la función **ivr_prebnfex**.

En caso de querer preparar una gramática en un canal determinado (generalmente, gramáticas que cambian con el tiempo y deben prepararse justo antes de ser utilizadas), debe utilizarse la función **ivr_prebnfdev**.

27.2. Descripción de los parámetros

filename	nombre del fichero que contiene la gramática `ABNF` a preparar.

27.3. Errores

`EVX_NOMEM`	OUT OF MEMORY
`EVX_VCBFILE`	THE VOCABULARY FILE NAME IS NOT VALID. Check the vocabulary file name and path writing permission.
`EVX_PARSER`	ABNF PARSER ERROR. Check the ABNF grammar syntax.
`EVX_INVWORD`	THE VOCABULARY CONTAINS AN INVALID WORD. Check and correct invalid words En caso de que se produzca este error, la función **ivr_errorline** proporcionará la línea que contiene la línea incorrecta del vocabulario.
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

27.4. Elementos relacionados

ivr_prevcbex, ivr_prevcb, ivr_prevcbdev, ivr_prebnfex, ivr_prebnfdev, ivr_errorline, ivr_setvcb, ivr_loadvcb

27.5. Códigos de ejemplo

Consulte el ejemplo.

28. ivr_prebnfex

FUNCION `ivr_prebnfex(filename, language)`

`filename`	nombre del fichero que contiene la gramática `ABNF` a preparar
`language`	idioma en el que se preparará la gramática, independientemente del idioma por defecto vigente

RETORNA 0 si el resultado es satisfactorio

-1 en caso de error

28.1. Descripción

La función **ivr_prebnfex()** compila la gramática especificada. El resultado de la compilación se almacena en un nuevo fichero de texto con el mismo nombre que la gramática pero con extensión `".trx"` (conteniendo el vocabulario de la gramática) y en un nuevo fichero de texto con extensión `".grm"` (conteniendo los estados de la gramática). También se realiza la trancripción fonética que se almacena en un nuevo fichero binario con el mismo nombre que el de la gramática pero con extensión `".trc"`. Esta función es independiente del canal y debe utilizarse una vez para cada gramática aunque ésta se utilice en distintos canales (sí es necesario invocar de nuevo a la función cuando se modifica dinámicamente el contenido de la gramática). En caso que deba preparar gramáticas dinámicas en distintos canales, debe utilizarse la función **ivr_prebnfdev** para evitar conflictos entre ellos.

En caso de querer preparar un vocabulario en un canal determinado (generalmente, vocabularios que cambian con el tiempo y deben prepararse justo antes de ser utilizados), debe utilizarse la función **ivr_prevcbdev**.

28.2. Descripción de los parámetros

filename nombre del fichero que contiene la gramática `ABNF` a preparar.

language permite especificar el idioma en el que debe prepararse la gramática `ABNF`. De este modo, no debe modificarse continuamente el idioma por defecto del reconocedor en aquellos entornos en los que se trabaje con vocabularios de múltiples idiomas. Los posibles valores son:

`"es"`	Español castellano
`"ca"`	Catalán
`"eu"`	Euskera
`"ga"`	Gallego
`"es-ar"`	Español argentino
`"es-cl"`	Español chileno
`"es-co"`	Español colombiano
`"es-mx"`	Español mexicano
`"es-ve"`	Español venezolano
`"pt"`	Portugués
`"pt-br"`	Portugués brasileño

`"fr"`	Francés
`"en-us"`	Inglés

28.3. Errores

`EVX_NOMEM`	OUT OF MEMORY
`EVX_VCBFILE`	THE VOCABULARY FILE NAME IS NOT VALID. Check the vocabulary file name and path writing permission.
`EVX_PARSER`	ABNF PARSER ERROR. Check the ABNF grammar syntax.
`EVX_INVWORD`	THE VOCABULARY CONTAINS AN INVALID WORD. Check and correct invalid words En caso de que se produzca este error, la función **ivr_errorline** proporcionará la línea que contiene la línea incorrecta del vocabulario.
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

28.4. Elementos relacionados

ivr_prebnf, ivr_prebnfdev, ivr_prevcb, ivr_prevcbex, ivr_prevcbdev, ivr_errorline, ivr_setvcb, ivr_loadvcb

28.5. Códigos de ejemplo

Consulte el ejemplo.

29. ivr_prebnfdev

FUNCION `ivr_prebnfdev`(`chdev`, `filename`, `language`)

`chdev`	descriptor de un canal abierto de Dialogic
`filename`	nombre del fichero que contiene la gramática `ABNF` a preparar
`language`	idioma en el que se preparará la gramática, independientemente del idioma por defecto vigente

RETORNA 0 si el resultado es satisfactorio

-1 en caso de error

29.1. Descripción

La función **ivr_prebnfdev()** compila la gramática especificada *en el canal indicado*. El resultado de la compilación se almacena en un nuevo fichero de texto con el mismo nombre que la gramática pero con extensión `".trx"` (conteniendo el vocabulario de la gramática) y en un nuevo fichero de texto con extensión `".grm"` (conteniendo los estados de la gramática). También se realiza la trancripción fonética que se almacena en un nuevo fichero binario con el mismo nombre que el de la gramática pero con extensión `".trc"`. Esta función está indicada para preparar gramáticas dinámicas que no pueden haberse preparado al lanzar la aplicación mediante las funciones **ivr_prebnf** o **ivr_prebnfex**.

Para evitar conflictos entre gramáticas preparadas simultáneamente entre distintos canales, es *absolutamente imprescindible* evitar que canales distintos preparen gramáticas con el mismo nombre. En caso de que se produjera este hecho, los resultados son imprevisibles. Por lo tanto, una estrategia recomendada es la de utilizar nombres de gramática que incluyan también el identificador del canal. De este modo, se imposibilita la coincidencia de nombres de gramática en canales distintos.

29.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta telefónica.
filename	nombre del fichero que contiene la gramática `ABNF` a preparar.
language	permite especificar el idioma en el que debe prepararse la gramática `ABNF`. De este modo, no debe modificarse continuamente el idioma por defecto del reconocedor en aquellos entornos en los que se trabaje con vocabularios de múltiples idiomas. Los posibles valores son:

`"es"`	Español castellano
`"ca"`	Catalán
`"eu"`	Euskera
`"ga"`	Gallego
`"es-ar"`	Español argentino
`"es-cl"`	Español chileno
`"es-co"`	Español colombiano
`"es-mx"`	Español mexicano

`"es-ve"`	Español venezolano
`"pt"`	Portugués
`"pt-br"`	Portugués brasileño
`"fr"`	Francés
`"en-us"`	Inglés

29.3. Errores

`EVX_NOMEM`	OUT OF MEMORY
`EVX_VCBFILE`	THE VOCABULARY FILE NAME IS NOT VALID. Check the vocabulary file name and path writing permission.
`EVX_PARSER`	ABNF PARSER ERROR. Check the ABNF grammar syntax.
`EVX_INVWORD`	THE VOCABULARY CONTAINS AN INVALID WORD. Check and correct invalid words En caso de que se produzca este error, la función **ivr_errorline** proporcionará la línea que contiene la línea incorrecta del vocabulario.
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

29.4. Elementos relacionados

ivr_prebnf, ivr_prebnfex, ivr_prevcb, ivr_prevcbex, ivr_prevcbdev, ivr_errorline, ivr_setvcb, ivr_loadvcb

29.5. Códigos de ejemplo

Consulte el ejemplo.

30. ivr_prevcb

FUNCION	`ivr_prevcb(filename)`	
	`filename`	nombre del fichero que contiene el vocabulario a preparar
RETORNA	0 si el resultado es satisfactorio	
	-1 en caso de error	

30.1. Descripción

La función **ivr_prevcb()** realiza la transcripción fonética del contenido del fichero, almacenándola en un nuevo fichero binario con el mismo nombre que el del vocabulario pero con extensión `".trc"`. Esta función es independiente del canal y debe utilizarse una vez para cada vocabulario aunque éste se utilice en distintos canales (sí es necesario invocar de nuevo a la función cuando se modifica dinámicamente el contenido del vocabulario). En caso que deba preparar vocabularios dinámicos en distintos canales, debe utilizarse la función **ivr_prevcbdev** para evitar conflictos entre ellos.

El vocabulario se compila siguiendo las reglas fonéticas del idioma establecido por defecto al inicializar el motor de reconocimiento, al invocar a la función **ivr_libinit** o **ivr_asr_init** o bien al modificar el parámetro `"0x80000403"` mediante la función **ivr_setparm**. En caso de querer preparar el vocabulario en un idioma conocido independiente del idioma por defecto, puede utililzarse la función **ivr_prevcbex**.

En caso de querer utilizar gramáticas `ABNF` (recomendado por su mayor flexibilidad), deben transcribirse mediante la función **ivr_prebnf()** o **ivr_prebnfex()**.

En caso de querer preparar un vocabulario en un canal determinado (generalmente, vocabularios que cambian con el tiempo y deben prepararse justo antes de ser utilizados), debe utilizarse la función **ivr_prevcbdev**.

30.2. Descripción de los parámetros

filename	nombre del fichero que contiene el vocabulario (cada línea constituye una `palabra del vocabulario` formada por una o más palabras) a preparar.

30.3. Errores

`EVX_NOMEM`	OUT OF MEMORY
`EVX_VCBFILE`	THE VOCABULARY FILE NAME IS NOT VALID. Check the vocabulary file name and path writing permission.
`EVX_PARSER`	ABNF PARSER ERROR. Check the ABNF grammar syntax.
`EVX_INVWORD`	THE VOCABULARY CONTAINS AN INVALID WORD. Check and correct invalid words En caso de que se produzca este error, la función **ivr_errorline** proporcionará la línea que contiene la línea incorrecta del vocabulario.
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

30.4. Elementos relacionados

ivr_prevcbex, ivr_prevcbdev, ivr_prebnf, ivr_prebnfex, ivr_prebnfdev, ivr_errorline, ivr_setvcb, ivr_loadvcb

30.5. Códigos de ejemplo

Consulte el ejemplo.

31. ivr_prevcbex

FUNCION `ivr_prevcbex`(`filename`, `language`)

`filename`	nombre del fichero que contiene el vocabulario a preparar
`language`	idioma en el que se preparará el vocabulario, independientemente del idioma por defecto vigente

RETORNA 0 si el resultado es satisfactorio

-1 en caso de error

31.1. Descripción

La función **ivr_prevcbex()** realiza la transcripción fonética del contenido del fichero, almacenándola en un nuevo fichero binario con el mismo nombre que el del vocabulario pero con extensión `".trc"`. Esta función es independiente del canal y debe utilizarse una vez para cada vocabulario aunque éste se utilice en distintos canales (sí es necesario invocar de nuevo a la función cuando se modifica dinámicamente el contenido del vocabulario). En caso que deba preparar gramáticas dinámicas en distintos canales, debe utilizarse la función **ivr_prevcbdev** para evitar conflictos entre ellos.

En caso de querer utilizar gramáticas ABNF (recomendado por su mayor flexibilidad), deben transcribirse mediante la función **ivr_prebnf()** o **ivr_prebnfex()**.

En caso de querer preparar un vocabulario en un canal determinado (generalmente, vocabularios que cambian con el tiempo y deben prepararse justo antes de ser utilizados), debe utilizarse la función **ivr_prevcbdev**.

31.2. Descripción de los parámetros

filename — nombre del fichero que contiene el vocabulario (cada línea constituye una `palabra del vocabulario` formada por una o más palabras) a preparar.

language — permite especificar el idioma en el que debe prepararse el vocabulario. De este modo, no debe modificarse continuamente el idioma por defecto del reconocedor en aquellos entornos en los que se trabaje con vocabularios de múltiples idiomas. Los posibles valores son:

`"es"`	Español castellano
`"ca"`	Catalán
`"eu"`	Euskera
`"ga"`	Gallego
`"es-ar"`	Español argentino
`"es-cl"`	Español chileno
`"es-co"`	Español colombiano
`"es-mx"`	Español mexicano
`"es-ve"`	Español venezolano
`"pt"`	Portugués
`"pt-br"`	Portugués brasileño

`"fr"`	Francés
`"en-us"`	Inglés

31.3. Errores

`EVX_NOMEM`	OUT OF MEMORY
`EVX_VCBFILE`	THE VOCABULARY FILE NAME IS NOT VALID. Check the vocabulary file name and path writing permission.
`EVX_PARSER`	ABNF PARSER ERROR. Check the ABNF grammar syntax.
`EVX_INVWORD`	THE VOCABULARY CONTAINS AN INVALID WORD. Check and correct invalid words En caso de que se produzca este error, la función **ivr_errorline** proporcionará la línea que contiene la línea incorrecta del vocabulario.
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

31.4. Elementos relacionados

ivr_prevcb, ivr_prevcbdev, ivr_prebnf, ivr_prebnfex, ivr_prebnfdev, ivr_errorline, ivr_setvcb, ivr_loadvcb

31.5. Códigos de ejemplo

Consulte el ejemplo.

32. ivr_prevcbdev

FUNCION `ivr_prevcbdev(chdev, filename, language)`

`chdev`	descriptor de un canal abierto de Dialogic
`filename`	nombre del fichero que contiene el vocabulario a preparar
`language`	idioma en el que se preparará el vocabulario, independientemente del idioma por defecto vigente

RETORNA 0 si el resultado es satisfactorio

-1 en caso de error

32.1. Descripción

La función **ivr_prevcbdev()** realiza la transcripción fonética del contenido del fichero *en el canal indicado*, almacenándola en un nuevo fichero binario con el mismo nombre que el del vocabulario pero con extensión `".trc"`. Esta función está indicada para preparar gramáticas dinámicas que no pueden haberse preparado al lanzar la aplicación mediante las funciones **ivr_prevcb** o **ivr_prevcbex**.

En caso de querer utilizar gramáticas ABNF (recomendado por su mayor flexibilidad), deben transcribirse mediante la función **ivr_prebnf()** o **ivr_prebnfex()** o **ivr_prebnfdev()**.

Para evitar conflictos entre vocabularios preparados simultáneamente entre distintos canales, es *absolutamente imprescindible* evitar que canales distintos preparen vocabularios con el mismo nombre. En caso de que se produjera este hecho, los resultados son imprevisibles. Por lo tanto, una estrategia recomendada es la de utilizar nombres de vocabulario que incluyan también el identificador del canal. De este modo, se imposibilita la coincidencia de nombres de vocabulario en canales distintos.

32.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta telefónica.
filename	nombre del fichero que contiene el vocabulario (cada línea constituye una `palabra del vocabulario` formada por una o más palabras) a preparar.
language	permite especificar el idioma en el que debe prepararse el vocabulario. De este modo, no debe modificarse continuamente el idioma por defecto del reconocedor en aquellos entornos en los que se trabaje con vocabularios de múltiples idiomas. Los posibles valores son:

`"es"`	Español castellano
`"ca"`	Catalán
`"eu"`	Euskera
`"ga"`	Gallego
`"es-ar"`	Español argentino
`"es-cl"`	Español chileno
`"es-co"`	Español colombiano
`"es-mx"`	Español mexicano

`"es-ve"`	Español venezolano
`"pt"`	Portugués
`"pt-br"`	Portugués brasileño
`"fr"`	Francés
`"en-us"`	Inglés

32.3. Errores

`EVX_NOMEM`
OUT OF MEMORY

`EVX_VCBFILE`
THE VOCABULARY FILE NAME IS NOT VALID. Check the vocabulary file name and path writing permission.

`EVX_PARSER`
ABNF PARSER ERROR. Check the ABNF grammar syntax.

`EVX_INVWORD`
THE VOCABULARY CONTAINS AN INVALID WORD. Check and correct invalid words

En caso de que se produzca este error, la función **ivr_errorline** proporcionará la línea que contiene la línea incorrecta del vocabulario.

`EVX_SYSTEM`
SYSTEM ERROR. Check errno.

32.4. Elementos relacionados

ivr_prevcb, ivr_prevcbex, ivr_prebnf, ivr_prebnfex, ivr_prebnfdev, ivr_errorline, ivr_setvcb, ivr_loadvcb

32.5. Códigos de ejemplo

Consulte el ejemplo.

33. ivr_reccfg_release

FUNCION `ivr_reccfg_release`(`chdev`, `config`)

`chdev`	descriptor de un canal abierto de Dialogic
`config`	configuración de la que se desea liberar una licencia o "" para licencia de motor ASR

RETORNA 0 si el resultado es satisfactorio

-1 en caso de error

33.1. Descripción

La función **ivr_reccfg_release()** libera un recurso (licencia) de reconocimiento del habla reservado mediante la función **ivr_reccfg_wait()** en el canal especificado. De este modo, otras peticiones de licencias pendientes podrán ser atendidas por el recurso liberado.

33.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

config Especifica el tipo de licencia de reconocimiento que debe liberarse. Los posibles valores son:

`""`	Libera una licencia de motor de reconocimiento
`"es"`	Español castellano
`"es_ca"`	Español castellano + catalán
`"es_eu"`	Español castellano + euskera
`"es_ga"`	Español castellano + gallego
`"es_ca_eu_ga"`	Español castellano + catalán + euskera + gallego
`"es-ar"`	Español argentino
`"es-cl"`	Español chileno
`"es-co"`	Español colombiano
`"es-mx"`	Español mexicano
`"es-ve"`	Español venezolano
`"pt"`	Portugués
`"pt-br"`	Portugués brasileño
`"fr"`	Francés
`"en-us"`	Inglés

33.3. Elementos relacionados

ivr_reccfg_wait, ivr_recstr_release, ivr_recstr_wait, ivr_playstr_release, ivr_playstr_wait

33.4. Códigos de ejemplo

Consulte el ejemplo.

34. ivr_reccfg_wait

FUNCION `ivr_reccfg_wait`(`chdev`, `ms`, `config`)

`chdev`	descriptor de un canal abierto de Dialogic
`ms`	tiempo máximo de espera para obtener un recurso (en milisegundos)
`config`	configuración de la que se desea reservar una licencia o "" para licencia de motor ASR

RETORNA 0 si el resultado es satisfactorio

-1 en caso de no poder obtener un recurso en el tiempo especificado

34.1. Descripción

La función **ivr_reccfg_wait()** es muy útil para controlar que, en el momento de requerir una licencia de reconocimiento (ya sea de motor o de configuración), se dispondrá de una, evitándose de este modo el error en las funciones invocadas con posterioridad. En la `Guía del Usuario` se detallan los momentos en los que se consume una licencia aunque, a grandes rasgos, podría resumirse del siguiente modo:

1. Las licencias de motor se consumen durante el procesado de las muestras con la finalidad de obtener la palabra o palabras del vocabulario reconocidas. Por norma general, el intervalo de uso de este tipo de licencias es pequeño.

2. Las licencias de configuración se empiezan a consumir desde que se establece un vocabulario en un canal o se manipula algún parámetro relacionado con el reconocimiento en un canal (`NBEST, ACTIVE_NBEST, KLENGTH, GRMWEIGHT, GRMCONSTANT, RECMODE, DEACTWORD y ACTWORD`) hasta que se cambia a otra configuración o se libera la licencia con la función miembro **ivr_reccfg_release()**. Lógicamente, el tiempo de uso de estas licencias es superior al de las licencias de motor (el vocabulario debe haberse establecido con anterioridad y no puede liberarse hasta finalizado el reconocimiento), llegando incluso (según la programación) a estar ocupadas durante gran parte de la ejecución.

En aquellos entornos que dispongan de tantas licencias como líneas atendidas, no es necesario tomar ningún tipo de precaución, de modo que el uso de estas funciones no aporta ninguna ventaja. Sin embargo, para evitar errores de funcionamiento debidos a falta de licencia en momentos puntuales en aquellos entornos que atiendan más líneas que licencias disponen, sí es muy aconsejable tener la precaución de reservar con antelación una licencia de motor y/o de configuración. De este modo, si no se obtiene la licencia en el momento deseado, se podrá mantener al usuario a la espera de forma controlada (por ejemplo, haciendo sonar música ambiental hasta que se disponga del recurso).

La función **ivr_reccfg_wait()** reserva un recurso (licencia) de reconocimiento del habla para el canal especificado hasta que se libere mediante la función **ivr_reccfg_release()**. La función espera un máximo de `ms` milisegundos la disponibilidad de un recurso, transcurridos los cuales devuelve un error.

34.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
ms	tiempo máximo de espera, en milisegundos, para obtener un recurso (licencia) de reconocimiento del habla para el canal especificado.
config	Especifica el tipo de licencia de reconocimiento que debe reservarse. Los posibles valores son:

`""`	Reserva una licencia de motor de reconocimiento

`"es"`	Español castellano
`"es_ca"`	Español castellano + catalán
`"es_eu"`	Español castellano + euskera
`"es_ga"`	Español castellano + gallego
`"es_ca_eu_ga"`	Español castellano + catalán + euskera + gallego
`"es-ar"`	Español argentino
`"es-cl"`	Español chileno
`"es-co"`	Español colombiano
`"es-mx"`	Español mexicano
`"es-ve"`	Español venezolano
`"pt"`	Portugués
`"pt-br"`	Portugués brasileño
`"fr"`	Francés
`"en-us"`	Inglés

34.3. Elementos relacionados

ivr_reccfg_release, ivr_recstr_wait, ivr_recstr_release, ivr_playstr_wait, ivr_playstr_release

34.4. Códigos de ejemplo

Consulte el ejemplo.

35. ivr_recfile

FUNCION	`ivr_recfile`(`chdev`, `mode`, `timeout`, `filename`)	
	`chdev`	descriptor de un canal abierto de Dialogic
	`mode`	uso futuro
	`timeout`	tiempo máximo, en décimas de segundo, que puede durar el proceso de grabación
	`filename`	nombre del fichero en el que se almacenarán las muestras de audio
RETORNA	0 si el resultado es satisfactorio	
	1 si la grabación ha finalizado por haber detectado un tono DTMF	
	7 si la grabación ha finalizado por haberse completado el tiempo máximo (timeout)	
	-1 en caso de error	

35.1. Descripción

La función **ivr_recfile()** inicia la grabación de la señal de voz presente en el canal especificado, en formato VOX a 8 KHz, 8 bits/muestra PCM. Una vez invocado, el proceso de grabación continua hasta que se detengan las operaciones en el canal o se cumpla alguna de las condiciones de terminación: recepción de un tono DTMF o finalización del tiempo máximo de grabación.

35.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
mode	uso futuro. Debe especificarse el valor 0.
timeout	tiempo máximo (expresado en décimas de segundo) que puede durar el proceso de grabación. Transcurrido este tiempo, la función finaliza.
filename	nombre del fichero en el que se almacenarán las muestras de audio.

35.3. Errores

`EVX_SRERROR`	DIALOGIC SRL ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_DXERROR`	DIALOGIC VOICE ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

35.4. Elementos relacionados

ivr_recstrf, ivr_recstr

35.5. Códigos de ejemplo

Consulte el ejemplo.

36. ivr_recind

FUNCION	`ivr_recind`(`chdev`, `maxind`)	
	`chdev`	descriptor de un canal abierto de Dialogic
	`maxind`	número máximo de `palabras del vocabulario` conectadas o de palabras de la gramática
RETORNA	0 si no se ha reconocido nada. En este caso, la función **ivr_id** devolverá el motivo.	
	>0 para indicar la cantidad de palabras reconocidas.	
	-1 en caso de error	

36.1. Descripción

La función **ivr_recind()** debe ser llamada al finalizar las funciones **ivr_recstr()** o **ivr_recstrf()** para obtener los índices y niveles de confianza de las palabras reconocidas.

En caso de querer conocer el resultado asociado a una hipótesis distinta de la hipótesis principal, debe especificarse previamente la hipótesis de interés mediante la función **ivr_selhyp**. El número de hipótesis de reconocimiento que generará el sistema de reconocimiento está fijado por el parámetro `"0x60000112"` y limitado por el parámetro `"0x60000101"`. Ambos parámetros pueden consultarse o modificarse mediante las funciones **ivr_getparm** y **ivr_setparm** respectivamente.

36.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
maxind	especifica el número máximo de `palabras del vocabulario` conectadas o palabras de la gramática que pueden reconocerse. En reconocimiento de palabras aisladas, el valor de esta parámetro debe fijarse a 1.

36.3. Errores

`EVX_NORECSTR`	NO RECOGNITION. Use **ivr_recstr()** to start recognition.

36.4. Elementos relacionados

ivr_recstr, ivr_recstrf, ivr_recresult, ivr_recscore

36.5. Códigos de ejemplo

Consulte el ejemplo.

37. ivr_recresult

FUNCION `ivr_recresult( chdev, rulename)`

`chdev`	descriptor de un canal abierto de Dialogic
`rulename`	nombre de la regla de la gramática ABNF activa de la que se quiere obtener el conjunto de palabras reconocidas. Si el valor es "", la función devuelve todas las palabras reconocidas

RETORNA vector de caracteres con todas las palabras de la regla reconocidas

vector vacío en caso de error

37.1. Descripción

La función **ivr_recresult()** obtiene todas las palabras reconocidas asociadas a la regla especificada de la gramática ABNF activada o bien el conjunto de todas las palabras reconocidas. Esta función debe invocarse siempre después de que finalice correctamente la función **ivr_recstr** o **ivr_recstrf**.

En caso de querer conocer el resultado asociado a una hipótesis distinta de la hipótesis principal, debe especificarse previamente la hipótesis de interés mediante la función **ivr_selhyp**. El número de hipótesis de reconocimiento que generará el sistema de reconocimiento está fijado por el parámetro `"0x60000112"` y limitado por el parámetro `"0x60000101"`. Ambos parámetros pueden consultarse o modificarse mediante las funciones **ivr_getparm** y **ivr_setparm** respectivamente.

37.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
varname	nombre de la regla de la gramática de la que se desea obtener la secuencia de palabras reconocida. En caso de querer obtener toda la cadena reconocida, debe especificarse el valor "".

37.3. Precauciones

La función **ivr_recresult()** debe ejecutarse siempre después de que finalice correctamente la función **ivr_recstr** o **ivr_recstrf**. No debe invocarse previamente a la función **ivr_recind**

37.4. Elementos relacionados

ivr_recstr, ivr_recstrf, ivr_recscore

37.5. Códigos de ejemplo

Consulte el ejemplo.

38. ivr_recscore

FUNCION `ivr_recscore( chdev, rulename)`

`chdev`	descriptor de un canal abierto de Dialogic
`rulename`	nombre de la regla de la gramática ABNF activa de la que se quiere obtener el índice de confianza del reconocimiento. Si el valor es "", la función devuelve el índice de confianza del resultado global.

RETORNA índice de confianza promediado de la regla especificada

-1 en caso de error

38.1. Descripción

La función **ivr_recscore()** obtiene el índice de confianza asociado a la regla especificada de la gramática ABNF activada o bien el índice de confianza global del resultado del reconocimiento. Esta función debe invocarse siempre después de que finalice correctamente la función **ivr_recstr** o **ivr_recstrf**.

En caso de querer conocer el resultado asociado a una hipótesis distinta de la hipótesis principal, debe especificarse previamente la hipótesis de interés mediante la función **ivr_selhyp**. El número de hipótesis de reconocimiento que generará el sistema de reconocimiento está fijado por el parámetro `"0x60000112"` y limitado por el parámetro `"0x60000101"`. Ambos parámetros pueden consultarse o modificarse mediante las funciones **ivr_getparm** y **ivr_setparm** respectivamente.

38.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
varname	nombre de la regla de la gramática de la que se desea obtener el índice de confianza En caso de querer obtener el índice global, debe especificarse el valor "".

38.3. Precauciones

La función **ivr_recscore()** debe ejecutarse siempre después de que finalice correctamente la función **ivr_recstr** o **ivr_recstrf**. No debe invocarse previamente a la función **ivr_recind**

38.4. Elementos relacionados

ivr_recstr, ivr_recstrf, ivr_recresult, ivr_score

38.5. Códigos de ejemplo

Consulte el ejemplo.

39. ivr_recstr

FUNCION `ivr_recstr`(`chdev`, `mode`, `timeout`, `initsil`, `maxsil`)

`chdev`	descriptor de un canal abierto de Dialogic
`mode`	especifica si debe reproducrise un tono antes de iniciarse el reconocimiento
`timeout`	tiempo máximo, en décimas de segundo, que puede durar el proceso de reconocimiento
`initsil`	silencio máximo, en décimas de segundo, antes de detectar voz
`maxsil`	silencio máximo, en décimas de segundo, después de detectar voz

RETORNA 0 si el resultado es satisfactorio

1 si el reconocimiento ha finalizado por haber detectado un tono DTMF

7 si el reconocimiento ha finalizado por haberse completado el tiempo máximo (timeout)

-1 en caso de error

39.1. Descripción

La función **ivr_recstr()** inicia el reconocimiento de la señal de voz presente en el canal especificado. Una vez invocado, el proceso de reconocimiento continua hasta que se detengan las operaciones en el canal o se cumpla alguna de las condiciones de terminación: recepción de un tono DTMF o cumplimiento de alguna condición temporal especificada mediante los parámetros.

39.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
mode	si se especifica "1", se reproducirá un tono antes de iniciarse el reconocimiento. En caso de especificar "0", no se reproduce ningún tono (`beep`).
timeout	tiempo máximo (expresado en décimas de segundo) que puede durar el proceso de reconocimiento. Transcurrido este tiempo, la función finaliza.
initsil	tiempo máximo (en décimas de segundo) de silencio que el reconocedor espera antes de detectar voz. Si no se ha detectado voz transcurrido este intervalo, la función finaliza.
maxsil	tiempo máximo (en décimas de segundo) de silencio que el reconocedor espera después de detectar voz. Una vez detectado este intervalo de silencio, la función finaliza.

39.3. Precauciones

La función **ivr_setvcb()** o **ivr_loadvcb()** debe ser llamada previamente a la función **ivr_recstr()** para establecer el vocabulario o gramática activo.

39.4. Errores

`EVX_SRERROR`	DIALOGIC SRL ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_DXERROR`	DIALOGIC VOICE ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().

`EVX_SYSTEM`	SYSTEM ERROR. Check errno.
`EVX_NOSETVCB`	NO ACTIVE VOCABULARY. Use **ivr_setvcb()** to set the active vocabulary.

39.5. Elementos relacionados

ivr_recstrf, ivr_recfile

39.6. Códigos de ejemplo

Consulte el ejemplo.

40. ivr_recstrf

FUNCION `ivr_recstrf`(`chdev`, `mode`, `timeout`, `initsil`, `maxsil`, `filename`)

`chdev`	descriptor de un canal abierto de Dialogic
`mode`	especifica si debe reproducrise un tono antes de iniciarse el reconocimiento
`timeout`	tiempo máximo, en décimas de segundo, que puede durar el proceso de reconocimiento
`initsil`	silencio máximo, en décimas de segundo, antes de detectar voz
`maxsil`	silencio máximo, en décimas de segundo, después de detectar voz
`filename`	nombre del fichero en el que se almacenarán las muestras recibidas

RETORNA 0 si el resultado es satisfactorio

1 si el reconocimiento ha finalizado por haber detectado un tono DTMF

7 si el reconocimiento ha finalizado por haberse completado el tiempo máximo (timeout)

-1 en caso de error

40.1. Descripción

La función **ivr_recstrf()** inicia el reconocimiento de la señal de voz y la almacena simultáneamente en el fichero especificado. Una vez invocado, el proceso de reconocimiento continua hasta que se detengan las operaciones en el canal o se cumpla alguna de las condiciones de terminación: recepción de un tono DTMF o cumplimiento de alguna condición temporal especificada mediante los parámetros.

40.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
mode	si se especifica "1", se reproducirá un tono antes de iniciarse el reconocimiento. En caso de especificar "0", no se reproduce ningún tono (`beep`).
timeout	tiempo máximo (expresado en décimas de segundo) que puede durar el proceso de reconocimiento. Transcurrido este tiempo, la función finaliza.
initsil	tiempo máximo (en décimas de segundo) de silencio que el reconocedor espera antes de detectar voz. Si no se ha detectado voz transcurrido este intervalo, la función finaliza.
maxsil	tiempo máximo (en décimas de segundo) de silencio que el reconocedor espera después de detectar voz. Una vez detectado este intervalo de silencio, la función finaliza.
filename	nombre del fichero en el que se almacenarán las muestras de audio recibidas por el reconocedor.

40.3. Precauciones

La función **ivr_setvcb()** o **ivr_loadvcb()** debe ser llamada previamente a la función **ivr_recstrf()** para establecer el vocabulario o gramática activo.

40.4. Errores

`EVX_SRERROR`	DIALOGIC SRL ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_DXERROR`	DIALOGIC VOICE ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.
`EVX_NOSETVCB`	NO ACTIVE VOCABULARY. Use **ivr_setvcb()** to set the active vocabulary.

40.5. Elementos relacionados

ivr_recstr, ivr_recfile

40.6. Códigos de ejemplo

Consulte el ejemplo.

41. ivr_recstr_release

FUNCION `ivr_recstr_release(chdev)`

`chdev` descriptor de un canal abierto de Dialogic

RETORNA 0 si el resultado es satisfactorio

-1 en caso de error

41.1. Descripción

La función **ivr_recstr_release()** libera un recurso (licencia) de reconocimiento del habla reservado mediante la función **ivr_recstr_wait()** en el canal especificado.

41.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

41.3. Elementos relacionados

ivr_recstr_wait, ivr_getusedasrlic, ivr_reccfg_release, ivr_reccfg_wait, ivr_playstr_release, ivr_playstr_wait

41.4. Códigos de ejemplo

Consulte el ejemplo.

42. ivr_recstr_wait

FUNCION `ivr_recstr_wait(chdev, ms)`

`chdev`	descriptor de un canal abierto de Dialogic
`ms`	tiempo máximo de espera para obtener un recurso de reconocimiento (en milisegundos)

RETORNA 0 si el resultado es satisfactorio

-1 en caso de no poder obtener un recurso en el tiempo especificado

42.1. Descripción

La función **ivr_recstr_wait()** reserva un recurso (licencia) de reconocimiento del habla para el canal especificado hasta que se libere mediante la función **ivr_recstr_release()**. La función espera un máximo de `ms` milisegundos la disponibilidad de un recurso, transcurridos los cuales devuelve un error.

El uso de esta función es aconsejable para gestionar el uso de los recursos (licencias) disponibles. En entornos en los que nunca se producirán más peticiones simultáneas de reconocimiento que licencias disponibles no es necesario su uso.

42.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
ms	tiempo máximo de espera, en milisegundos, para obtener un recurso (licencia) de reconocimiento del habla para el canal especificado.

42.3. Elementos relacionados

ivr_recstr_release, ivr_getusedasrlic, ivr_reccfg_wait, ivr_reccfg_release, ivr_playstr_wait, ivr_playstr_release

42.4. Códigos de ejemplo

Consulte el ejemplo.

43. ivr_score

FUNCION `ivr_score( chdev, index)`

`chdev`	descriptor de un canal abierto de Dialogic
`index`	número de la palabra devuelta por el reconocedor de la que se desea conocer el índice de confianza

RETORNA índice de confianza de la palabra especificada si el resultado es satisfactorio

-1 en caso de error

43.1. Descripción

La función **ivr_score()** devuelve el índice de confianza de la palabra especificada como parámetro. Esta función debe invocarse siempre después de que finalice correctamente la función **ivr_recstr** o **ivr_recstrf**.

En caso de querer conocer el resultado asociado a una hipótesis distinta de la hipótesis principal, debe especificarse previamente la hipótesis de interés mediante la función **ivr_selhyp**. El número de hipótesis de reconocimiento que generará el sistema de reconocimiento está fijado por el parámetro `"0x60000112"` y limitado por el parámetro `"0x60000101"`. Ambos parámetros pueden consultarse o modificarse mediante las funciones **ivr_getparm** y **ivr_setparm** respectivamente.

43.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

index Número de la palabra devuelta por el reconocedor de la que se quiere obtener el resultado. El valor de este parámetro debe estar comprendido entre 0 y el número de palabras devuelto por el reconocedor, indicado como resultado de la función **ivr_recind()**.

43.3. Elementos relacionados

ivr_recstr, ivr_recstrf, ivr_recind, ivr_recscore

43.4. Códigos de ejemplo

Consulte el ejemplo.

44. ivr_selhyp

FUNCION `ivr_selhyp( chdev, hyp)`

`chdev`	descriptor de un canal abierto de Dialogic
`hyp`	número de la hipótesis de la que se quiere obtener el resultado de reconocimiento

RETORNA 0 si el resultado es satisfactorio

-1 en caso de error

44.1. Descripción

La función **ivr_selhyp()** especifica la hipótesis de interés de la que se desea obtener el resultado del reconocimiento mediante la función **ivr_recind()**. La cantidad de hipótesis deseadas se fija mediante la función **ivr_setparm** y el parámetro `"0x60000112"`antes de lanzar el proceso de reconocimiento.

44.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

hyp número de la hipótesis de la que se quiere obtener el resultado de reconocimiento mediante la función **ivr_recind()**. El valor especificado debe ser siempre menor que el fijado mediante el parámetro `"0x60000112"` de la función **ivr_setparm**.

44.3. Elementos relacionados

ivr_setparm, ivr_recind

44.4. Códigos de ejemplo

Consulte el ejemplo.

45. ivr_setabbreviations

FUNCION	`ivr_setabbreviations(chdev, mcLang, filename)`	
	`chdev`	descriptor de un canal abierto de Dialogic
	`mcLang`	especifica el idioma asociado al diccionario personal de abreviaturas
	`filename`	puntero al nombre del fichero de texto que contiene las abreviaturas personalizadas
RETORNA	0 si el resultado es satisfactorio	
	-1 en caso de error	

45.1. Descripción

La función **ivr_setabbreviations()** establece un conjunto de abreviaturas personalizadas para ser utilizadas por el servidor de síntesis en el canal especificado, pudiendo ser modificadas dinámicamente.

El diccionario de abreviaturas personalizadas se añade al diccionario global de abreviaturas usado por el servidor de síntesis para el idioma especificado.

Para una mayor información, consulte el capítulo de síntesis del habla en `Verbio Software Reference: Guía del Usuario`.

45.2. Descripción de los parámetros

chdev — especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

mclang — especifica el idioma asociado con el vocabulario de abreviaturas personalizadas.

`"es"`	Español castellano
`"ca"`	Catalán
`"eu"`	Euskera
`"ga"`	Gallego
`"es-mx"`	Español mexicano
`"pt"`	Portugués
`"pt-br"`	Portugués brasileño*
`"fr"`	Francés*
`"en-us"`	Inglés*

(*) consultar las condiciones particulares

filename — puntero al nombre del fichero de texto que contiene las abreviaturas personalizadas para el idioma seleccionado.

45.3. Elementos relacionados

ivr_setdictionary

45.4. Códigos de ejemplo

Consulte el ejemplo.

46. ivr_setcd

FUNCION	`ivr_setcd(chdev)`	
	`chdev`	descriptor de un canal abierto de Dialogic
RETORNA	0 si el resultado es satisfactorio	
	-1 en caso de error	

46.1. Descripción

La función **ivr_setcd()** establece el vocabulario de dígitos conectados en el canal especificado.

46.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

46.3. Precauciones

Si se establece el vocabulario o la gramática activa mediante la función **ivr_setvcb()** o **ivr_setcd()**, se descargarán automáticamente todos los vocabularios y gramáticas cargados hasta ese momento.

46.4. Errores

`EVX_NOMEM`	OUT OF MEMORY
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

46.5. Elementos relacionados

ivr_setvcb, ivr_loadvcb, ivr_activatevcb, ivr_deactivatevcb, ivr_unloadvcb, ivr_ivcb

47. ivr_setdictionary

FUNCION `ivr_setdictionary(chdev, mcLang, filename)`

`chdev`	descriptor de un canal abierto de Dialogic
`mcLang`	especifica el idioma asociado al diccionario personal
`filename`	puntero al nombre del fichero de texto que contiene el diccionario personalizado

RETORNA 0 si el resultado es satisfactorio

-1 en caso de error

47.1. Descripción

La función **ivr_setdictionary()** establece un diccionario personalizado para ser utilizado por el servidor de síntesis en el canal especificado, pudiendo ser modificado dinámicamente.

El diccionario personalizado se añade al diccionario global usado por el servidor de síntesis para el idioma especificado.

Para una mayor información, consulte el capítulo de síntesis del habla en `Verbio Software Reference: Guía del Usuario`.

47.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

mclang especifica el idioma al que afectará el diccionario personalizado.

`"es"`	Español castellano
`"ca"`	Catalán
`"eu"`	Euskera
`"ga"`	Gallego
`"es-mx"`	Español mexicano
`"pt"`	Portugués
`"pt-br"`	Portugués brasileño*
`"fr"`	Francés*
`"en-us"`	Inglés*

(*) consultar las condiciones particulares

filename puntero al nombre del fichero de texto que contiene el diccionario personalizado para el idioma seleccionado.

47.3. Elementos relacionados

ivr_setabbreviations

47.4. Códigos de ejemplo

Consulte el ejemplo.

48. ivr_setdtmf

FUNCION `ivr_setdtmf(chdev, dtmf)`

`chdev`	descriptor de un canal abierto de Dialogic
`dtmf`	tonos DTMF que podrán interrumpir la siguiente reproducción de síntesis

RETORNA 0 si el resultado es satisfactorio

-1 en caso de error

48.1. Descripción

La función **ivr_setdtmf()** permite especificar qué tonos DTMF pueden interrumpir únicamente la *siguiente* reproducción (es decir, una vez finalizada la siguiente reproducción, se volverá al comportamiento por defecto). El comportamiento por defecto interrumpe la síntesis en caso de que se pulse cualquier tono DTMF, de modo que es imprescindible el uso de esta función en cuanto se desee modificar este comportamiento.

48.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

dtmf especifica el tono o los tonos DTMF que podrán interrumpir la siguiente reproducción:

`""`	ningún tono DTMF interrumpirá la siguiente reproducción
`"1"`	tecla 1
`"2"`	tecla 2
`"3"`	tecla 3
`"4"`	tecla 4
`"5"`	tecla 5
`"6"`	tecla 6
`"7"`	tecla 7
`"8"`	tecla 8
`"9"`	tecla 9
`"*"`	tecla * (star)
`"#"`	tecla # (pound)
`"A"`	tecla A
`"B"`	tecla B
`"C"`	tecla C
`"D"`	tecla D

48.3. Elementos relacionados

ivr_getdtmf

48.4. Códigos de ejemplo

Para no interrumpir la siguiente grabación, *ivr_setdtmf(line, "")*

Para interrumpir la siguiente grabación si se pulsa un 1 o un 2, *ivr_setdtmf(line, "12")*

Para interrumpir la siguiente grabación con cualquier tecla, déjese el comportamiento por defecto, sin invocar a esta función

49. ivr_setparm

FUNCION `ivr_setparm`(`chdev`, `parm`, `value`)

`chdev` descriptor de un canal abierto de Dialogic o -1 para parámetros globales de `VERBIO` (independientes del canal)

`parm` parámetro del que se desea establecer el valor

`value` valor del parámetro a modificar

RETORNA 0 si el resultado es satisfactorio

-1 en caso de error

49.1. Descripción

La función **ivr_setparm()** permite modificar los valores de los parámetros globales o dependientes del canal de *VERBIO*. Esta función puede modificar el valor de un único parámetro en cada invocación.

49.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic. Use `-1` para establecer el valor de parámetros globales de *VERBIO* (independientes del canal).

parm especifica el identificador del parámetro del que se desea modificar el valor. Para conocer los identificadores válidos, consulte el Capítulo 3. *Estructuras de datos y parámetros*.

value valor del parámetro a modificar.

49.3. Elementos relacionados

ivr_getparm

49.4. Códigos de ejemplo

Consulte el ejemplo.

50. ivr_setvcb

FUNCION	`ivr_setvcb(chdev, filename, mode)`	
	`chdev`	descriptor de un canal abierto de Dialogic
	`filename`	puntero al nombre del fichero que contiene el vocabulario o la gramática `ABNF` o `""` para liberar todos los vocabularios o gramáticas y licencias de idiomas
	`mode`	vocabulario convencional o gramática `ABNF`
RETORNA	0 si el resultado es satisfactorio	
	-1 en caso de error	

50.1. Descripción

La función **ivr_setvcb()** establece el vocabulario o la gramática `ABNF` especificado en el canal especificado.

50.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
filename	nombre del fichero que contiene el vocabulario (cada línea constituye una `palabra del vocabulario` formada por una o más palabras) o la gramática `ABNF`, o `""` para liberar todos los vocabularios o gramáticas y licencias de idiomas.
mode	indica el tipo de vocabulario o gramática.

`"0x00010000"`	Palabras aisladas. El reconocedor espera reconocer una única `palabra del vocabulario` de este vocabulario.
`"0x00000000"`	Palabras conectadas (valor por defecto). El reconocedor espera reconocer una o más `palabras del vocabulario` de este vocabulario.
`"0x00080000"`	gramática `ABNF`. El reconocedor espera reconocer una o más palabras de la gramática.

50.3. Precauciones

Sin la opción `"0x00080000"` habilitada, la función **ivr_setvcb()** requiere el fichero de definiciones y el fichero de transcripciones generados mediante la función **ivr_prevcb()** (o **ivr_prevcbex()** o **ivr_prevcbdev()**). Si estos ficheros existen y el fichero de vocabulario correspondiente no ha sido modificado desde que se llamó a la función **ivr_prevcb()** (o **ivr_prevcbex()** o **ivr_prevcbdev()**), no es necesario volver a invocarlas antes de cargar el vocabulario mediante la función **ivr_setvcb()**.

Con la opción `"0x00080000"` habilitada, la función **ivr_setvcb()** requiere el fichero de estados de la gramática, el fichero de definiciones y el fichero de transcripciones generado mediante la función **ivr_prebnf()** (o **ivr_prebnfex()** o **ivr_prebnfdev()**). Si estos ficheros existen y la gramática `ABNF` correspondiente no ha sido modificado desde que se llamó a la función **ivr_prebnf()** (o **ivr_prebnfex()** o **ivr_prebnfdev()**), no es necesario volver a invocarlas antes de cargar la gramática mediante la función **ivr_setvcb()**.

Si se establece el vocabulario o la gramática activa mediante la función **ivr_setvcb()** o **ivr_setcd()**, se descargarán automáticamente todos los vocabularios y gramáticas cargados hasta ese momento.

En caso de que se especifique el valor `""` como parámetro `filename`, se descargarán todos los vocabularios cargados (equivale a invocar a la función **ivr_unloadvcb** con el valor a `-1`) y, además, *se liberarán las licencias de idioma ocupadas por ese canal*.

50.4. Errores

`EVX_NOMEM`	OUT OF MEMORY.
`EVX_VCBFILE`	THE VOCABULARY FILE NAME IS NOT VALID.
`EVX_INVWORD`	THE VOCABULARY TRANSCRIPTION WAS NOT FOUND.
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

50.5. Elementos relacionados

ivr_setcd, ivr_loadvcb, ivr_activatevcb, ivr_deactivatevcb, ivr_unloadvcb, ivr_ivcb

50.6. Códigos de ejemplo

Consulte el ejemplo.

51. ivr_asv_addfile

FUNCION int `ivr_asv_addfile`(int `chdev`, const char* `id`, const char* `filename`, const char* `transcription`)

int `chdev`	descriptor de un canal abierto de Dialogic
const char *`id`	identificador del usuario que entrena su huella vocal con el fichero de audio indicado
const char *`filename`	nombre del fichero de audio utilizado durante la fase de entreno del usuario indicado
const char *`transcription`	transcripción del contenido del audio utilizado para entrenar la huella vocal del usuario indicado

RETORNA 0 si el resultado es satisfactorio

<0 en caso de error

51.1. Descripción

La función **ivr_spk_addfile()** almacena un fichero de audio en la base de datos de usuarios para ser utilizado en el momento de crear la huella vocal del usuario indicado, mediante la invocación de la función **ivr_spk_train**. Adicionalmente, utilizando la capacidad de reconocimiento de *Verbio ASR*, comprueba también que el contenido del fichero coincide con la transcripción, de modo que pueda certificarse que el usuario ha pronunciado exactamente aquello que se le ha solicitado.

51.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
id	identificador del usuario del que se ha obtenido un nuevo fichero de audio para el proceso de entrenamiento. el identificador de un usuario es una cadena alfanumérica (dígitos o caracteres a-z,A-Z) elegida por el desarrollador de la aplicación. Debe ser única para cada usuario, siendo habitual que el identificador coincida con el identificador numérico (cadena de dígitos) que se le solicitará al usuario para identificarse en el sistema.
filename	nombre completo del fichero de audio que contiene la locución del usuario indicado, destinado a ser utilizado en el proceso de creación de la huella vocal. El formato del fichero debe ser WAV (8000 muestras/segundo, Ley A, Ley Mu o PCM lineal).
transcription	transcripción del contenido del fichero de audio que será utilizado en el proceso de creación de la huella vocal, para certificar que el contenido coincide con la secuencia de dígitos solicitada.

51.3. Precauciones

El contenido de los ficheros de entrenamiento debe ser, exclusivamente, cadenas de dígitos (de longitud variable). Por lo tanto, la transcripción proporcionada deben ser cadenas numéricas, del tipo `12345678`, `952611232`, etcétera. Se aconseja que las cadenas de entrenamiento tengan una longitud mínima de 8 dígitos y que se proporcionen, al menos, 20 de ellas, obtenidas en distintos días (mínimo dos) y momentos del día (mañana/tarde).

Puede invocarse directamente a la función **ivr_asv_train** especificando un listado de ficheros a utilizar para la fase de entrenamiento en lugar de invocar, para cada nuevo fichero, a la función **ivr_asv_addfile**. No obstante, se aconseja su uso a medida que se obtienen los ficheros porque en caso de errores (debido a que el usuario no ha pronunciado la cadena de dígitos solicitada) podrá notificarse este error al usuario y solicitarle un nuevo fichero.

51.4. Errores

En caso de error, la función retorna un valor negativo. Los posibles códigos de error retornados por esta función son:

`-1` El canal especificado no es válido.

`-4` Error interno de Verbio. La estructura de la base de datos está corrupta.

`-6` Error al lanzar el proceso de reconocimiento del contenido del fichero de audio.

`-18` Algún parámetro no ha sido especificado o bien es incorrecto.

`-27` El contenido del fichero no coincide con la transcripción proporcionada, o bien no se ha podido comprobar.

51.5. Elementos relacionados

ivr_asv_init, ivr_asv_exist, ivr_asv_train, ivr_asv_verify

51.6. Códigos de ejemplo

Consulte el ejemplo anexo.

52. ivr_asv_exist

FUNCION int `ivr_asv_exist`(int `chdev`, const char* `id`)

int `chdev`	descriptor de un canal abierto de Dialogic
const char `*id`	identificador del usuario del que se desea conocer su existencia (registro previo)

RETORNA 2 si el usuario está registrado y dispone de huella vocal

1 si el usuario está registrado y no dispone de huella vocal

0 si el usuario no está registrado

<0 en caso de error

52.1. Descripción

La función **ivr_asv_exist()** consulta la base de datos de usuarios registrados para comprobar si ya ha completado la fase de entrenamiento y, por lo tanto, ya dispone de un modelo (huella vocal) creado.

52.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

id identificador del usuario del que se quiere conocer el estado de su proceso de registro.

el identificador de un usuario es una cadena alfanumérica (dígitos o caracteres a-z,A-Z) elegida por el desarrollador de la aplicación. Debe ser única para cada usuario, siendo habitual que el identificador coincida con el identificador numérico (cadena de dígitos) que se le solicitará al usuario para identificarse en el sistema.

52.3. Errores

Si la función retorna -1 indicando que se ha producido un error, debe comprobarse que el canal especificado como primer parámetro es válido.

52.4. Elementos relacionados

ivr_asv_init, ivr_asv_addfile, ivr_asv_train, ivr_asv_verify

52.5. Códigos de ejemplo

Consulte el ejemplo anexo.

53. ivr_asv_init

FUNCION	int `ivr_asv_init()`
RETORNA	0 si el resultado es satisfactorio
	<0 en caso de error

53.1. Descripción

La función **ivr_asv_init()** inicializa la librería `Verbio`, preparándola para poder recibir peticiones de verificación de locutor.

53.2. Precauciones

La función **ivr_asv_init()** debe invocarse previamente a cualquier invocación del resto de funciones asociadas a la verificación de locutor, tales como: **ivr_asv_addfile**, **ivr_asv_train**, **ivr_asv_verify** y **ivr_asv_exist**.

53.3. Errores

En caso de error, la función retorna un valor negativo. Los posibles códigos de error retornados por esta función son:

`-13` Error interno de Verbio. Probablemente la instalación de los componentes de ASV no se ha realizado o se ha hecho de forma incorrecta.

`-14` No se dispone de licencias de ASV o bien éstas han caducado.

53.4. Elementos relacionados

ivr_asv_addfile, ivr_asv_train, ivr_asv_verify, ivr_asv_exist

53.5. Códigos de ejemplo

Consulte el ejemplo anexo.

54. ivr_asv_train

FUNCION	int `ivr_asv_train`(int `chdev`, const char* `id`, const char* `filename`)

int `chdev`	descriptor de un canal abierto de Dialogic
const char *`id`	identificador del usuario que entrena su huella vocal con el fichero de audio indicado
const char *`filename`	nombre del fichero que contiene el listado de ficheros a utilizar durante el proceso de entrenamiento

RETORNA 0 si el resultado es satisfactorio

<0 en caso de error

54.1. Descripción

La función **ivr_spk_train()** lanza el proceso de generación de la huella vocal del usuario indicado. Las locuciones del usuario que se utilizarán para este proceso serán las especificadas mediante el parámetro `filename` junto con todas aquellas asignadas al usuario anteriormente mediante llamadas a las funciones **ivr_spk_addfile** y/o **ivr_spk_train()**.

Una vez completada correctamente la función **ivr_spk_train()**, el usuario ya estará listo para que se realicen sobre él operaciones de verificación mediante la función **ivr_spk_verify()**.

En caso de que el usuario ya dispusiera de huella vocal, se generará una nueva huella vocal que reemplazará a la ya existente.

54.2. Descripción de los parámetros

chdev — especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.

id — identificador del usuario del que se desea obtener la huella vocal a partir de los ficheros de audio y transcripciones indicados.

el identificador de un usuario es una cadena alfanumérica (dígitos o caracteres a-z,A-Z) elegida por el desarrollador de la aplicación. Debe ser única para cada usuario, siendo habitual que el identificador coincida con el identificador numérico (cadena de dígitos) que se le solicitará al usuario para identificarse en el sistema.

filename — NULL si los ficheros de audio y transcripciones se han proporcionado invocando sucesivamente a la función **ivr_asv_addfile**.

nombre completo del fichero de texto que contiene el listado de ficheros de audio a utilizar para el entrenamiento junto con sus respectivas transcripciones. Los ficheros de audio deben ser WAV (8000 muestras/segundo, Ley A, Ley Mu o PCM lineal). El formato de este fichero debe ser el siguiente (una línea, acabada en retorno de carro, para cada fichero de audio a utilizar):

`nombre_fichero_audio` TAB `transcripcion_cadena_digitos` RC

RC = Retorno de Carro - TAB = Tabulador

54.3. Precauciones

El contenido de los ficheros de entrenamiento debe ser, exclusivamente, cadenas de dígitos (de longitud variable). Por lo tanto, la transcripción proporcionada deben ser cadenas numéricas, del tipo `12345678`,

`952611232`, etcétera. Se aconseja que las cadenas de entrenamiento tengan una longitud mínima de 8 dígitos y que se proporcionen, al menos, 20 de ellas, obtenidas en distintos días (mínimo dos) y momentos del día (mañana/tarde).

En lugar de especificar conjuntamente el listado de ficheros y sus transcripciones mediante la función **ivr_asv_train** es preferible utilizar la función **ivr_asv_addfile** individualmente para cada fichero. De este modo, podrán tomarse acciones correctivas en caso de que, en el momento de adquisición del audio, el proceso de validación interno indique que la transcripción no coincide con el contenido real del fichero.

54.4. Errores

En caso de error, la función retorna un valor negativo. Los posibles códigos de error retornados por esta función son:

`-1` El canal especificado no es válido.

`-4` Error interno de Verbio. La estructura de la base de datos está corrupta.

`-5` No se dispone de suficiente memoria disponible para ejecutar la función.

`-6` Error al lanzar el proceso de reconocimiento del contenido del fichero de audio.

`-10` Error al lanzar el proceso de verificación del locutor.

`-18` Algún parámetro no ha sido especificado o bien es incorrecto.

`-21` El fichero especificado como parámetro no está disponible.

`-27` El contenido de algún fichero no coincide con la transcripción proporcionada, o bien no se ha podido comprobar. Puede obtenerse la linea del fichero que contiene el audio inválido invocando a la función **ivr_errorlinedev(chdev)**.

`-28` No se dispone de suficiente material para entrenar. Es necesario proporcionar más locuciones numéricas del usuario a entrenar.

54.5. Elementos relacionados

ivr_asv_init, ivr_asv_exist, ivr_asv_addfile, ivr_asv_verify

54.6. Códigos de ejemplo

Consulte el ejemplo anexo.

55. ivr_asv_verify

FUNCION	int `ivr_asv_verify`(int `chdev`, const char* `id`, const char* `filename`, const char* `transcription`)	
	int `chdev`	descriptor de un canal abierto de Dialogic
	const char *`id`	identificador del usuario que verifica su identidad con el fichero de audio proporcionado
	const char *`filename`	nombre del fichero de audio utilizado para verificar la identidad del usuario.
	const char *`transcription`	transcripción del contenido del audio utilizado para verificar la identidad del usuario
RETORNA	1 si el usario es quien dice ser	
	0 si el usario NO es quien dice ser	
	<0 en caso de error	

55.1. Descripción

La función **ivr_asv_verify()** lanza el proceso de verificación para determinar si la locución proporcionada pertenece (ha sido pronunciada) realmente por el usuario indicado, basándose en la huella vocal que ha sido previamente entrenada para dicho usuario mediante la función **ivr_asv_train**.

55.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
id	identificador del usuario cuya identidad pretende validarse mediante su huella vocal. el identificador de un usuario es una cadena alfanumérica (dígitos o caracteres a-z,A-Z) elegida por el desarrollador de la aplicación. Debe ser única para cada usuario, siendo habitual que el identificador coincida con el identificador numérico (cadena de dígitos) que se le solicitará al usuario para identificarse en el sistema.
filename	nombre completo del fichero de audio que contiene la locución del usuario indicado, destinado a ser utilizado en el proceso de verificación de su huella vocal. El formato del fichero debe ser WAV (8000 muestras/segundo, Ley A, Ley Mu o PCM lineal).
transcription	transcripción del contenido del fichero de audio que será utilizado en el proceso de verificación de la huella vocal, para certificar que el contenido coincide con la secuencia de dígitos solicitada.

55.3. Precauciones

El contenido de los ficheros de verificación debe ser, exclusivamente, cadenas de dígitos (de longitud variable). Por lo tanto, la transcripción proporcionada deben ser cadenas numéricas, del tipo `12345678`, `952611232`, etcétera. Se aconseja que las cadenas de verificación tengan una longitud de entre 6 y 8 dígitos (cuanto más larga, mayor fiablidad del proceso de verificación).

El índice de confianza del proceso de verificación devuelto por la función **ivr_score(chdev, 1)** se proporciona a efectos estadísticos y de ajuste de los umbrales. El sistema, internamente, ya toma la decisión de si es el usuario verídico o no, retornando el valor 1 o 0 respectivamente como resultado de la función **ivr_asv_verify**.

55.4. Errores

En caso de error, la función retorna un valor negativo. Los posibles códigos de error retornados por esta función son:

`-1` El canal especificado no es válido.

`-4` Error interno de Verbio. La estructura de la base de datos está corrupta.

`-5` No se dispone de suficiente memoria disponible para ejecutar la función.

`-6` Error al lanzar el proceso de reconocimiento del contenido del fichero de audio.

`-10` Error al lanzar el proceso de verificación del locutor.

`-18` Algún parámetro no ha sido especificado o bien es incorrecto.

`-21` El fichero especificado como parámetro no está disponible.

`-27` El contenido del fichero no coincide con la transcripción proporcionada, o bien no se ha podido comprobar.

55.5. Elementos relacionados

ivr_asv_init, ivr_asv_exist, ivr_asv_train, ivr_asv_addfile

55.6. Códigos de ejemplo

Consulte el ejemplo anexo.

56. ivr_tts_init

FUNCION `ivr_tts_init( configuration, defttslng)`

`configuration`	uso futuro. Debe tener el valor "".
`defttslng`	identificador del idioma de síntesis por defecto

RETORNA 0 si el resultado es satisfactorio

-1 en caso de error

56.1. Descripción

La función **ivr_tts_init()** inicializa la librería de síntesis del habla de Verbio. En nuevos desarrollos es recomendable utilizar esta función (conjuntamente con la función **ivr_asr_init** en caso de que también se requiera reconocimiento del habla) en lugar de la función **ivr_libinit**, puesto que permite indicar al servidor los parámetros de inicialización de una manera mucho más concreta.

56.2. Descripción de los parámetros

configuration	parámetro destinado a usos futuros. El valor debe ser fijado a "".
defttslng	especifica el idioma por defecto con el que trabajará el servidor de síntesis.

`"es"`	Español castellano
`"ca"`	Catalán
`"eu"`	Euskera
`"ga"`	Gallego
`"es-mx"`	Español mexicano
`"pt"`	Portugués
`"pt-br"`	Portugués brasileño*
`"fr"`	Francés*
`"en-us"`	Inglés*

(*) consultar condiciones

56.3. Precauciones

La función **ivr_tts_init()** debe invocarse antes que ninguna otra de las funciones del SDK de Verbio relacionadas con la síntesis del habla, a excepción de aquella destinada a establecer el servidor de trabajo por defecto en configuraciones cliente-servidor (**ivr_setparm**) o bien a su equivalente en renococimiento (**ivr_asr_init**) .

56.4. Errores

`EVX_SRERROR`	DIALOGIC SRL ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().

EVX_DXERROR	DIALOGIC VOICE ERROR. Check ATDV_LASTERR() and ATDV_ERRMSGP().
EVX_NOBOARDS	NO BOARDS DETECTED. Check Dialogic Drivers Start Up.
EVX_INVSETUP	VERBIO INTERNAL ERROR. Files may be corrupted. Check disk and repeat Setup.
EVX_NOMEM	OUT OF MEMORY.
EVX_NOLICFILE	NO LICENSE FILE WAS FOUND. Run CheckOut to obtain a license file.
EVX_INVLIC	THE LICENSE FILE IS NOT VALID. Run CheckOut to obtain a valid license file.
EVX_SYSTEM	SYSTEM ERROR.Check errno.
EVX_INVSRVER	INVALID DIALOGIC SRL RELEASE (libsrlmt.dll). Check README.TXT.
EVX_INVDXVER	INVALID DIALOGIC VOICE LIBRARY RELEASE (libdxxmt.dll).

56.5. Elementos relacionados

ivr_libinit, ivr_asr_init

56.6. Códigos de ejemplo

Consulte el ejemplo.

57. ivr_unloadvcb

FUNCION	`ivr_unloadvcb(chdev, vcbhandle)`	
	`chdev`	descriptor de un canal abierto de Dialogic
	`vcbhandle`	descriptor de un vocabulario o gramática ABNF previamente cargado
RETORNA	>=0 indicando el descriptor del vocabulario si el resultado es satisfactorio	
	-1 en caso de error	

57.1. Descripción

La función **ivr_unloadvcb()** descarga un vocabulario o gramática ABNF previamente cargado en el canal especificado. El descriptor del vocabulario se obtiene al cargarlo mediante la función **ivr_loadvcb()** (o **ivr_loadcd()**) y también debe usarse para activarlo mediante la función **ivr_activatevcb()** o para descactivarlo mediante la función **ivr_deactivatevcb()**. Si el descriptor del vocabulario es `-1`, la función **ivr_unloadvcb()** descarga todos los vocabularios y gramáticas ABNF cargados previamente.

57.2. Descripción de los parámetros

chdev	especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta Dialogic.
vcbhandle	descriptor de un vocabulario o gramática ABNF previamente cargado mediante la función **ivr_loadvcb()**.

57.3. Precauciones

Si se establece el vocabulario o la gramática activa mediante la función **ivr_setvcb()** o **ivr_setcd()**, se descargarán automáticamente todos los vocabularios y gramáticas cargados hasta ese momento (equivale a invocar la función **ivr_unloadvcb()** con el parámetro `vcbhandle` fijado al valor `-1`).

A diferencia de invocar a la función **ivr_setvcb** con el parámetro `filename` con valor `""`, la función **ivr_unloadvcb** con el parámetro `vcbhandle` con valor `-1` *no libera la licencia de idioma consumida por el canal*.

57.4. Errores

`EVX_NOMEM`	OUT OF MEMORY
`EVX_VCBFILE`	THE VOCABULARY FILE NAME IS NOT VALID.
`EVX_INVWORD`	THE VOCABULARY TRANSCRIPTION WAS OT FOUND
`EVX_SYSTEM`	SYSTEM ERROR. Check errno.

57.5. Elementos relacionados

ivr_setvcb, ivr_loadvcb, ivr_loadcd, ivr_activatevcb, ivr_deactivatevcb, ivr_ivcb

57.6. Códigos de ejemplo

Consulte el ejemplo.

58. ivr_version

FUNCION	`ivr_version( lib, type)`	
	`lib`	librería de la que se desea conocer la versión
	`type`	información que se quiere obtener de la versión
RETORNA	vector con la versión si el resultado es satisfactorio	
	vector vacío en caso de error	

58.1. Descripción

La función **ivr_version()** devuelve la versión de la librería especificada. Es importante que la versión de la `RLL` para `CT ADE` coincida con la versión de *Verbio*.

58.2. Descripción de los parámetros

lib — librería de la que se desea conocer la versión.

`"0"`	`RLL` para `CT ADE`
`"1"`	versión de *Verbio*

type — información que se quiere obtener de la versión.

`"0"`	versión del producto
`"1"`	versión del fichero

58.3. Elementos relacionados

ivr_libinit, ivr_asr_init, ivr_tts_init

58.4. Códigos de ejemplo

Consulte el ejemplo.

59. ivr_word

FUNCION `ivr_word(chdev, ind)`

`chdev` descriptor de un canal abierto de Dialogic

`ind` índice asociado con la palabra del vocabulario o gramática que se desea obtener

RETORNA vector de caracteres con el contenido de la palabra del vocabulario apuntada por el índice especificado

vector vacío `("")` si el índice especificado no es válido

59.1. Descripción

La función **ivr_word()** devuelve un vector de caracteres ASCIIZ con el contenido asociado al índice (fila) especificado del vocabulario o gramática activa en el canal.

En caso de querer conocer el resultado asociado a una hipótesis distinta de la hipótesis principal, debe especificarse previamente la hipótesis de interés mediante la función **ivr_selhyp**. El número de hipótesis de reconocimiento que generará el sistema de reconocimiento está fijado por el parámetro `"0x60000112"` y limitado por el parámetro `"0x60000101"`. Ambos parámetros pueden consultarse o modificarse mediante las funciones **ivr_getparm** y **ivr_setparm** respectivamente.

59.2. Descripción de los parámetros

chdev especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta telefónica.

ind índice asociado con la palabra del vocabulario o gramática que se desea obtener. Generalmente, los índices de interés son aquellos devueltos por la función **ivr_id**. En el caso de querer obtener toda la información posible ofrecida por el servidor de reconocimiento (reglas, valores, locuciones, niveles de confianza e intervalos temporales), debe utilizarse el valor `IND_STREAM`, en cuyo caso el vector de caracteres ASCIIZ contiene toda la información estructurada siguiendo el formato especificado en Sección 59.5.

59.3. Elementos relacionados

ivr_wordex, ivr_recstr, ivr_recstrf, ivr_id

59.4. Códigos de ejemplo

Consulte el ejemplo.

59.5. Obtención de información adicional mediante ivr_word

El servidor de reconocimiento, a parte de la palabra o palabras reconocidas y del nivel de confianza de las mismas, puede proporcionar más datos que pueden resultar de interés en algunos casos: reglas, valores , locuciones, niveles de confianza y tiempos. Antes de proceder con la explicación de cada uno de estos elementos, es importante que el lector se familiarice con los aspectos relacionados con los vocabularios convencionales y las gramáticas `ABNF`, que encontrará detallados en *Verbio Software Reference: Guía del usuario*. A continuación se dará una explicación del significado de cada uno de estos campos para poder hacer uso de ellos en caso de considerarse necesario.

Tabla 2.1. Elementos de un vocabulario o gramática ABNF

Elemento	Descripción
Regla	En vocabularios convencionales no se especifica explícitamente, de modo que se le asignará siempre el valor `root`. En gramáticas ABNF las reglas ayudan a diseñar, compartir y clarificar el contenido de las mismas. En el ejemplo siguiente, existen 3 reglas: *color* (regla principal), *rgb* y *comp*.
Valor	Los valores son los `items` que devuelve el servidor como resultado del reconocimiento. En vocabularios convencionales suele ser el contenido de la primera columna. En el ejemplo inferior, alguno de los posibles valores devueltos son: *Comercial* o *Técnico*. En gramáticas ABNF los valores se especifican entre llaves si difieren de la palabra del vocabulario. En el ejemplo siguiente, alguno de los posibles valores devueltos son: *COLOR=R* o *color* pero no *rojo* porque se le ha asociado explícitamente el valor *COLOR=R*.
Locución	Las locuciones se corresponden con lo que el usuario realmente ha dicho. En vocabularios convencionales suele ser el contenido de la segunda columna (si sólo hay una columna, valor y locución coinciden). En el ejemplo inferior, alguna de las posibles locuciones devueltas son: *Eduardo Campillo* o *Asunción Chinea*. En gramáticas ABNF son el texto que no tenga asociado ningún valor vacío {}. En el ejemplo siguiente, alguna de las posibles locuciones devueltas son: *rojo* o *color* pero no *COLOR=R* (se ha especificado como valor) ni *por favor* (se le ha asociado el valor vacío {}).
Confianza	El nivel de confianza es un valor numérico que indica el grado de fiabilidad que se le da al resultado. A mayor valor, mayor probabilidad de que la secuencia reconocida sea la correcta.
Intervalo	El intervalo de tiempo en el que se ha pronunciado la locución es otro de los datos ofrecidos por el servidor de reconocimiento. Concretamente, proporciona el instante de inicio y el instante de finalización de la locución.

El formato del vector de caracteres ASCIIZ devuelto por la función **ivr_word()** es el siguiente:

regla:"locución"(confianza,instante_inicial,instante_final)["valor1" "valor2" "..."]

En caso de no tratarse de una gramática ABNF, el nombre de la regla será `root`. En caso de tratarse de una gramática ABNF con subreglas (no únicamente una regla principal), el formato del vector de caracteres tendrá un formato parecido al siguiente, en el que cada regla termina en `";"` y cada subregla está contenida entre llaves `"{}"`, y en el que debe sustituirse "regla" por todo el texto asociado a cada regla (según se ha especificado en el párrafo anterior).

regla_principal{regla1{regla1_1;regla1_2;};regla2;regla3{regla3_1{regla3_1_1;};};};

La gramática ABNF asociada al ejemplo anterior podría ser parecida a:

```
#ABNF 1.0 ISO8859-1;
root regla_principal = ... $regla1 ... $regla2 ... $regla3;

$regla1 = ... $regla1_1 ... $regla1_2 ...;
$regla1_1 = ... ;
$regla1_2 = ... ;

$regla2 = ... ;

$regla3 = ... $regla3_1 ... ;
$regla3_1 = ... $regla3_1_1 ... ;
$regla3_1_1 = ... ;
```

Vocabulario convencional.

```
Comercial           Eduardo Campillo
Comercial           Asunción Chinea
Técnico             Marta Cuenca
Administración      Emilio Doblado
Administración      Guillermo Estrada
Técnico             Sergio Galíndez
```

Gramática ABNF de múltiples reglas.

```
#ABNF 1.0 ISO8859-1;
$rgb  = rojo   {COLOR=R} |      // se utiliza la palabra "rojo" y devuelve la palabra COLOR=R
        verde  {COLOR=G} |      // se utiliza la palabra "verde" y devuelve la palabra COLOR=G
        azul   {COLOR=B};       // se utiliza la palabra "verde" y devuelve la palabra COLOR=G

$comp = marino {COMP=D} |
        osuro  {COMP=D} |
        claro  {COMP=L};

root $color = [color | "el color" | el] $rgb  [$comp] ["por favor"{}];
```

Ejemplo 2.1. Información devuelta por el servidor de reconocimiento

Locución dictada: *"Eduardo Campillo"*

```
root:"Eduardo Campillo"(158.29,0.50,1.28)["Eduardo Campillo"];
```

Locución dictada: *"El color rojo por favor"*

```
color:"el color rojo"(127.59,0.26,1.00)["el color" "COLOR=R"]{rgb:"rojo"(168.40,0.76,1.00)["COLOR=R"];comp:;};
```

Locución dictada: *"El azul marino"*

```
color:"el azul marino"(185.46,0.56,1.37)["el" "COLOR=B" "COMP=D"]{rgb:"azul"(93.27,0.76,1.00)["COLOR=B"];
comp:"marino"(318.19,1.00,1.37)["COMP=D"];};
```

60. ivr_wordex

FUNCION `ivr_wordex(chdev, ind, column)`

`chdev`	descriptor de un canal abierto de Dialogic
`ind`	índice asociado con la palabra del vocabulario o gramática que se desea obtener
`column`	columna (información) devuelta por el reconocedor

RETORNA vector de caracteres con el contenido de la palabra del vocabulario apuntada por el índice especificado

vector vacío `("")` si el índice especificado no es válido

60.1. Descripción

La función **ivr_wordex()** devuelve un vector de caracteres ASCIIZ con el contenido asociado al índice (fila) y columna especificado del vocabulario o gramática activa en el canal.

En caso de querer conocer el resultado asociado a una hipótesis distinta de la hipótesis principal, debe especificarse previamente la hipótesis de interés mediante la función **ivr_selhyp**. El número de hipótesis de reconocimiento que generará el sistema de reconocimiento está fijado por el parámetro `"0x60000112"` y limitado por el parámetro `"0x60000101"`. Ambos parámetros pueden consultarse o modificarse mediante las funciones **ivr_getparm** y **ivr_setparm** respectivamente.

60.2. Descripción de los parámetros

chdev
: especifica el descriptor de canal válido obtenido al abrir el canal de la tarjeta telefónica.

ind
: índice asociado con la palabra del vocabulario o gramática que se desea obtener. Generalmente, los índices de interés son aquellos devueltos por la función **ivr_id**. En el caso de querer obtener toda la información posible ofrecida por el servidor de reconocimiento (reglas, valores, locuciones, niveles de confianza e intervalos temporales), debe utilizarse el valor `IND_STREAM`, en cuyo caso el vector de caracteres ASCIIZ contiene toda la información estructurada siguiendo el formato especificado en Sección 60.5.

column
: identificador de la columna de interés. En la columna `0` (por defecto, en la función **ivr_word**) se almacena el valor devuelto por la gramática. En la columna `1` se almacena la *utterance* o locución pronunciada por el usuario. En la columna `2` se almacena el peso asociado a la palabra en la gramática. Finalmente, en la columna `3` se almacena el nombre de la regla de la gramática a la que pertenece la palabra reconocida. Consulte *Verbio Software Reference: Guía del usuario* para más detalles acerca del contenido de cada una de las columnas que constituyen un vocabulario.

60.3. Elementos relacionados

ivr_word, ivr_recstr, ivr_recstrf, ivr_id

60.4. Códigos de ejemplo

Consulte el ejemplo.

60.5. Obtención de información adicional mediante ivr_word

El servidor de reconocimiento, a parte de la palabra o palabras reconocidas y del nivel de confianza de las mismas, puede proporcionar más datos que pueden resultar de interés en algunos casos: reglas, valores , locuciones, niveles de confianza y tiempos. Antes de proceder con la explicación de cada uno de estos elementos, es importante que el lector se familiarice con los aspectos relacionados con los vocabularios

convencionales y las gramáticas ABNF, que encontrará detallados en *Verbio Software Reference: Guía del usuario*. A continuación se dará una explicación del significado de cada uno de estos campos para poder hacer uso de ellos en caso de considerarse necesario.

Tabla 2.2. Elementos de un vocabulario o gramática ABNF

Elemento	Descripción
Regla	En vocabularios convencionales no se especifica explícitamente, de modo que se le asignará siempre el valor `root`. En gramáticas ABNF las reglas ayudan a diseñar, compartir y clarificar el contenido de las mismas. En el ejemplo siguiente, existen 3 reglas: *color* (regla principal), *rgb* y *comp*.
Valor	Los valores son los `items` que devuelve el servidor como resultado del reconocimiento. En vocabularios convencionales suele ser el contenido de la primera columna. En el ejemplo inferior, alguno de los posibles valores devueltos son: *Comercial* o *Técnico*. En gramáticas ABNF los valores se especifican entre llaves si difieren de la palabra del vocabulario. En el ejemplo siguiente, alguno de los posibles valores devueltos son: *COLOR=R* o *color* pero no *rojo* porque se le ha asociado explícitamente el valor *COLOR=R*.
Locución	Las locuciones se corresponden con lo que el usuario realmente ha dicho. En vocabularios convencionales suele ser el contenido de la segunda columna (si sólo hay una columna, valor y locución coinciden). En el ejemplo inferior, alguna de las posibles locuciones devueltas son: *Eduardo Campillo* o *Asunción Chinea*. En gramáticas ABNF son el texto que no tenga asociado ningún valor vacío {}. En el ejemplo siguiente, alguna de las posibles locuciones devueltas son: *rojo* o *color* pero no *COLOR=R* (se ha especificado como valor) ni *por favor* (se le ha asociado el valor vacío {}).
Confianza	El nivel de confianza es un valor numérico que indica el grado de fiabilidad que se le da al resultado. A mayor valor, mayor probabilidad de que la secuencia reconocida sea la correcta.
Intervalo	El intervalo de tiempo en el que se ha pronunciado la locución es otro de los datos ofrecidos por el servidor de reconocimiento. Concretamente, proporciona el instante de inicio y el instante de finalización de la locución.

El formato del vector de caracteres ASCIIZ devuelto por la función **ivr_word()** es el siguiente:

regla:"locución"(confianza,instante_inicial,instante_final)["valor1" "valor2" "..."]

En caso de no tratarse de una gramática ABNF, el nombre de la regla será `root`. En caso de tratarse de una gramática ABNF con subreglas (no únicamente una regla principal), el formato del vector de caracteres tendrá un formato parecido al siguiente, en el que cada regla termina en `";"` y cada subregla está contenida entre llaves `"{}"`, y en el que debe sustituirse "regla" por todo el texto asociado a cada regla (según se ha especificado en el párrafo anterior).

regla_principal{regla1{regla1_1;regla1_2;};regla2;regla3{regla3_1{regla3_1_1;};};};

La gramática ABNF asociada al ejemplo anterior podría ser parecida a:

```
#ABNF 1.0 ISO8859-1;
root regla_principal = ... $regla1 ... $regla2 ... $regla3;

$regla1 = ... $regla1_1 ... $regla1_2 ...;
$regla1_1 = ... ;
$regla1_2 = ... ;

$regla2 = ... ;

$regla3 = ... $regla3_1 ... ;
$regla3_1 = ... $regla3_1_1 ... ;
$regla3_1_1 = ... ;
```

Vocabulario convencional.

```
Comercial           Eduardo Campillo
Comercial           Asunción Chinea
Técnico             Marta Cuenca
Administración      Emilio Doblado
Administración      Guillermo Estrada
Técnico             Sergio Galíndez
```

Gramática ABNF de múltiples reglas.

```
#ABNF 1.0 ISO8859-1;
$rgb  = rojo   {COLOR=R} |      // se utiliza la palabra "rojo" y devuelve la palabra COLOR=R
        verde  {COLOR=G} |      // se utiliza la palabra "verde" y devuelve la palabra COLOR=G
        azul   {COLOR=B};       // se utiliza la palabra "verde" y devuelve la palabra COLOR=G

$comp = marino {COMP=D} |
        osuro  {COMP=D} |
        claro  {COMP=L};

root $color = [color | "el color" | el] $rgb  [$comp] ["por favor"{}];
```

Ejemplo 2.2. Información devuelta por el servidor de reconocimiento

Locución dictada: *"Eduardo Campillo"*

```
root:"Eduardo Campillo"(158.29,0.50,1.28)["Eduardo Campillo"];
```

Locución dictada: *"El color rojo por favor"*

```
color:"el color rojo"(127.59,0.26,1.00)["el color" "COLOR=R"]{rgb:"rojo"(168.40,0.76,1.00)["COLOR=R"];comp:;};
```

Locución dictada: *"El azul marino"*

```
color:"el azul marino"(185.46,0.56,1.37)["el" "COLOR=B" "COMP=D"]{rgb:"azul"(93.27,0.76,1.00)["COLOR=B"];
comp:"marino"(318.19,1.00,1.37)["COMP=D"];};
```



Capítulo 3. Estructuras de datos y parámetros

1. Definición de parámetros para ivr_getparm() y ivr_setparm()

A continuación se incluyen los parámetros que pueden ser consultados y modificados mediante las funciones **ivr_getparm()** y **ivr_setparm()** respectivamente. Existen parámetros globales (independientes del canal) y parámetros dependientes del canal.

Tabla 3.1. Parámetros globales en el SDK de CTADE

Define	Tipo	Lectura(R)/ Escritura(R)	Defecto	Descripción
"0x60800201"	int	R	-	Licencias de reconocimiento disponibles, i.e., número máximo de llamadas simultáneas a las funciones **ivr_recstr()** o **ivr_recstrf()**.
"0x60800202"	int	R	-	Licencias de síntesis disponibles, i.e., número máximo de llamadas simultáneas a la función **ivr_playstr()**.
"0xE0800203"	float	R	-	Frecuencia de muestreo a la que trabaja el motor de reconocimiento.
"0x60800206"	int	R/W	5	Intervalo de guarda (en segundos) entre las comunicaciones cliente-servidor. Transcurrido este intervalo, se produce un error para indicar que la comunicación entre el cliente y el servidor se ha perdido. En la fase de depuración es conveniente poner este valor a `-1` para evitar que se produzca un error al ejecutarse la aplicación paso a paso.
"0x80000801"	char*	R/W	"127.0.0.1"	Nombre o dirección IP del equipo en el que está operativo el servidor de síntesis y/o reconocimiento al que se conecta el cliente. Es necesario especificarlo antes de invocar a la función **ivr_libinit** o **ivr_asr_init** o **ivr_tts_init**.
"0xE0800204"	float	R/W	200	Gestión de la amplitud máxima permitida para señal de voz. Condiciona el comportamiento del detector de voz (VAD) del motor de reconocimiento.
"0xE0800207"	float	R/W	50	Gestión de la amplitud mínima permitida para señal de voz. Condiciona el comportamiento del detector de voz (VAD) del motor de reconocimiento. Para entornos ruidosos, es necesario incrementar el valor de este parámetro para que el ruido de fondo no se confunda por señal de voz. El valor de este parámetro no podrá nunca superar el especificado mediante el parámetro `VXGB_VSDMAXREF`.

Define	Tipo	Lectura(R)/ Escritura(R)	Defecto	Descripción
"0x80000802"	char*	R	-	Nombre o dirección IP del equipo en el que está ejecutándose el cliente.
"0x80000804"	char*	R	-	Listado de todos los locutores disponibles junto con sus características principales. El formato del vector de caracteres obtenido es el siguiente (contiene información de todos los locutores encontrados sin espacios intermedios): `Identificador: Nombre: Sexo: Edad: Idioma; Identificador2: Nombre2: Sexo2: Edad2: Idioma2; ...`
"0x60800208"	int	R/W	0	Habilita (1) o deshabilita (0) la grabación de todas las locuciones enviadas al reconocedor junto con los resultados de reconocimiento asociados. Es especialmente últil en la fase de pruebas para poder evaluar la calidad de reconocimiento y actuar, si cabe, sobre las gramáticas o los diálogos para mejorar el rendimiento.
"0xA0800209"	char*	R/W	""	Directorio en el que se almacenarán las locuciones y los resultados de reconocimiento si se ha habilitado el parámetro "0x60800208".
"0xA080020A"	char*	W	-	Especifica el modelo estadístico que será utilizado por la aplicación. Para ello, el valor del parámetro debe ser el nombre de la *arquitectura* que contiene los modelos deseados. Para más detalles acerca de cómo utilizar el reconocimiento basado en modelos estadísticos, consulte los ejemplos o contacte con support@verbio.com.
"0x80000805"	char*	R	-	Permite obtener el listado de locutores disponibles en el motor de síntesis para ser utilizados desde la aplicación. Esta función devuelve los códigos internos de los locutores, separados por comas. Los códigos internos de los locutores siguen la siguiente nomenclatura: languageNAME (identificador de idioma en minúscula seguido de dos letras mayúsculas que identifican al locutor). Para obtener un listado completo de los identificadores de idioma (de 2 o 5 caracteres), consulte la función ivr_tts_init.
"0x80000806"	char*	R	-	Permite obtener el listado de configuraciones de reconocimiento disponibles en el motor de

Define	Tipo	Lectura(R)/ Escritura(R)	Defecto	Descripción
				reconocimiento para ser utilizados desde la aplicación. Esta función devuelve los códigos internos de las configuraciones, separados por comas. Los códigos internos de las configuraciones siguen la siguiente nomenclatura: LANGUAGE (para configuraciones monolingües) o LANGUAGE-LANGUAGE.... -LANGUAGE (para configuraciones multilingües). Para obtener un listado completo de los identificadores de idioma (de 2 o 5 caracteres), consulte la función ivr_asr_init.
"0x80000807"	char*	R	-	Permite obtener el listado arquitecturas que contienen modelos estadísticos (SLM) disponibles para ser utilizadas desde el motor de reconocimiento estadístico. El listado contiene cada uno de los identificadores de las arquitecturas, separados por comas. El nombre de las arquitecturas es fijado por el integrador durante la construcción de los modelos estadísticos. Para más detalles acerca de cómo utilizar el reconocimiento basado en modelos estadísticos, consulte los ejemplos o contacte con support@verbio.com.
"0x80000808"	char*	R	-	Permite obtener el listado de modelos estadísticos (SLM) disponibles dentro de la arquitectura cargada para ser utilizadas desde el motor de reconocimiento estadístico. El listado contiene cada uno de los identificadores de los modelos estadísticos, separados por comas. El nombre de los modelos es fijado por el integrador durante la construcción de los mismos. Para más detalles acerca de cómo utilizar el reconocimiento basado en modelos estadísticos, consulte los ejemplos o contacte con support@verbio.com.
"0x60800209"	int	W	2500	Permite definir la longitud máxima de la respuesta ante un proceso de reconocimiento. El valor son la cantidad de unidades de 20ms permitidas a lo largo de todo el proceso, de modo que, con el valor por defecto, la longitud máxima de audio (antes de lanzar un nuevo reconocimiento) que puede procesarse es de 50 segundos. El valor de este parámetro es un entero, de modo que no debe ser superior a 32767.

Tabla 3.2. Parámetros de canal en el SDK de CTADE

Define	Tipo	Lectura(R)/ Escritura(R)	Defecto	Descripción
`"0x60000101"`	int	R/W	1	Número máximo de hipótesis generadas por las funciones de reconocimiento **ivr_recstr()** y **ivr_recstrf()**. Es muy importante tener en cuenta que una modificación de este parámetro implica tener que volver a cargar y activar todos los vocabularios y gramática previamente activos. Por lo tanto, es aconsejable fijar este valor desde un principio al valor máximo de hipótesis que vayan a requerirse durante el funcionamiento de toda la aplicación y, según las necesidades de cada momento, ir solicitando más o menos hipótesis mediante la modificación del parámetro `"0x60000112"`.
`"0x60000112"`	int	R/W	1	Número de hipótesis generadas por las funciones de reconocimiento **ivr_recstr()** y **ivr_recstrf()**. A diferencia del parámetro `"0x60000101"`, la modificación de este parámetro no afecta a los vocabularios y gramáticas activos, sino únicamente a la cantidad de hipótesis devueltas por los mismos. Sí hay que tener en cuenta, sin embargo, que nunca se podrán solicitar más hipótesis mediante este parámetro que las especificadas mediante el parámetro `"0x60000101"`.
`"0x60000113"`	int	R	-	Frecuencia de muestreo a la que trabaja el motor de síntesis.
`"0x60000104"`	int	R/W	-	Velocidad de la conversión de texto a voz en palabras por minuto. El valor por defecto depende de cada locutor.
`"0x80000402"`	char*	R/W	-	Nombre del locutor de síntesis activo. El valor inicial dependerá de cómo se haya configurado el servidor de síntesis mediante el *Verbio Server Configuration Manager*. Este mismo configurador permite ver el nombre de los locutores asociados a cada uno de los idiomas disponibles.
`"0x60000106"`	int	R/W	-	Especifica el idioma de síntesis mediante una constante que se usará en el próximo proceso de conversión de texto a habla. Se mantiene por compatibilidad con versiones anteriores.

Define	Tipo	Lectura(R)/ Escritura(R)	Defecto	Descripción
				Para nuevos desarrollos debería utilizarse el parámetro `"0x80000404"`. El valor inicial dependerá de cómo se haya configurado el servidor de síntesis mediante el *Verbio Server Configuration Manager*.
`"0x80000404"`	char*	R/W	-	Especifica el idioma de síntesis en formato texto que se usará en el próximo proceso de conversión de texto a habla. En nuevos desarrollos es la opción recomendada frente al obsoleto `"0x60000106"`. El valor inicial dependerá de cómo se haya configurado el servidor de síntesis mediante el *Verbio Server Configuration Manager*.
`"0x60000107"`	int	R/W	-	Tono medio de la voz sintetizada. Las voces femeninas tienen un tono superior que el de las voces masculinas. El valor por defecto depende del locutor.
`"0x6000010D"`	int	R/W	-	Volumen de la voz sintetizada. La amplitud se amplifica proporcionalmente a la relación entre el nuevo valor y el valor por defecto. Por lo tanto, la estrategia recomendada consiste en consultar el valor actual del locutor, manipular el valor proporcionalmente para obtener el volumen deseado y fijarlo posteriormente. Se desaconseja pues fijar un valor aleatorio sin conocer el valor actual de funcionamiento.
`"0x80000405"`	char*	R/W	-	Configuración de reconocimiento por defecto que se utilizará en cualquier proceso de reconocimiento hasta que se especifique lo contrario. El valor inicial dependerá de cómo se haya configurado el servidor de reconocimiento mediante el *Verbio Server Configuration Manager*.
`"0x80000403"`	char*	R/W	-	Idioma por defecto que usarán las funciones **ivr_prevcb()** y **ivr_prebnf()** para procesar los ficheros de vocabulario. El valor inicial dependerá de cómo se haya configurado el servidor de reconocimiento mediante el *Verbio Server Configuration Manager*.
`"0x80000401"`	char*	R	-	Servidor de síntesis y/o reconocimiento al que se conecta el canal (nombre o IP del equipo). El valor por defecto depende del parámetro `"0x80000801"`, aunque puede

Define	Tipo	Lectura(R)/ Escritura(R)	Defecto	Descripción
				modificarse con posterioridad para que cada línea pueda especificar su propio servidor.
"0x6000010F"	int	R	-	Indica si se ha detectado voz durante el proceso de reconocimiento (el valor devuelto en el parámatro es *1*). Típicamente la consulta de este parámetro se realiza para conocer el momento en el que el usuario ha empezado a hablar, de modo que puedan aplicarse, por ejemplo, estrategias de `Barge-in`.
"0x80000406"	char*	W	-	permite deshabilitar una regla de una gramática o de todas las gramáticas activas en un momento determinado. De este modo, en entornos en los que se parta de una gramática compleja que, en base al diálogo, pueda irse acotando, no es necesario desglosarla en gramáticas más sencillas a medida que el usuario navega las opciones, sino que pueden irse desactivando aquellas reglas que ya no aplican en la siguiente posible respuesta. Con ello se mejora la rapidez y la calidad del reconocimiento sin apenas coste computacional, puesto que no hay que descargar ni cargar nuevas gramáticas. El valor del parámetro debe ser el nombre de la regla de la gramática `ABNF` que debe desactivarse. En caso de que ese nombre pueda estar presentes en otras gramáticas activas en ese instante, puede especificarse el identificador de la gramática de interés mediante el siguiente formato: `nombre_regla#identificador`, donde `identificador` es el valor devuelto por la función **ivr_setvcb** o **ivr_activatevcb**. Si se desea desactivar esa regla en todas las gramáticas activas que la contengas, el parámetro debe contener únicamente el nombre de la regla.
"0x80000407"	char*	W	-	permite habilitar de nuevo una regla de una gramática o de todas las gramáticas activas en un momento determinado que hubiera sido desactivada, previamente, mediante `"0x80000406"`. El formato del parámetro debe ser idéntico al especificado en la descripción de `"0x80000406"`.
"0xE000010C"	float	R/W	55	Valor que regula el nivel de poda del motor de reconocimiento. Cuanto mayor es el valor, menos descartes

Define	Tipo	Lectura(R)/ Escritura(R)	Defecto	Descripción
				realiza el motor de reconocimiento, por lo que puede afinarse algo más el resultado de reconocimiento, aunque a costa de un incremento sustancial del coste computacional. No se aconseja modificar el valor de este parámetro, salvo consejo del personal de *Verbio*.
`"0x6000010E"`	int	R	-	Permite consultar los milisegundos de voz (excluído el silencio) que se han recibido en el anterior proceso de reconocimiento.
`"0x60000110"`	int	R	-	Permite consultar los milisegundos de audio que se han recibido en el anterior proceso de reconocimiento.

Capítulo 4. Código de ejemplo

Nota

El Apéndice A contiene los ficheros de vocabularios y gramáticas utilizados en los ejemplos de esta sección.

1. Ejemplo

Ejemplo 4.1. Ejemplo de Verbio CT ADE

verbio.vs

```
dec
        const IVR     = "0";
        const IBERVOX = "1";
        const PRODUCT = "0";
        const FILE    = "1";
        const IVXGB_RECSTRLIC  = "201";
        const IVXGB_PLAYSTRLIC = "202";
        # VoxServer
        const IVXGB_DEFSERVER  = "801";
        # IberVox Languages
        const LVX_SPANISH      = "0x00020000";
        const LVX_CATALAN      = "0x01000000";
enddec

program
        ivr_setparm(0, IVXGB_DEFSERVER, "192.168.0.9"); # If VoxServer Client
        # ivr_libinit(LVX_SPANISH);
        ivr_asr_init("es_ca", "es");
        ivr_tts_init("", "es");

        vid_write("IVR RLL ", ivr_version(IVR, PRODUCT));
        vid_write("Verbio ", ivr_version(IBERVOX, PRODUCT));

        #vid_write("TTS licenses: ", ivr_getparm(0, IVXGB_PLAYSTRLIC));
        #vid_write("ASR licenses: ", ivr_getparm(0, IVXGB_RECSTRLIC));
        vid_write("TTS licenses (Spanish): ", ivr_getttslic("es"));
        vid_write("TTS licenses (Catalan): ", ivr_getttslic("ca"));
        vid_write("ASR licenses (Spanish & Catalan): ", ivr_getttslic("es"));

        ivr_prevcb("nombres.txt");
        ivr_prebnf("fechas.bnf");
        Master("IberVoxVR");
endprogram
```

main.vs

```
dec
        const IND_NOWORD       = "-1000";
        # IberVox Languages
        const LVX_SPANISH      = "0x00000000";
        const LVX_CATALAN      = "0x01000000";
        # IberVox Grammar Types
        const IVR_CONNECTED    = "0"; # connected words
        const IVR_ISOLATED     = "1"; # isolated words
        const IVR_ABNF         = "8"; # ABNF
        # Channel parameters
        const IVXCH_TTSSPEED   = "104";
        const IVXCH_TTSLNG     = "106";
        const IVXCH_TTSPITCH   = "107";
        const IVXCH_DEFASRLNG  = "10C";
        const IVXCH_TTSVOLUME  = "10D";
        const IVXCH_TTSSPKNAME = "402";

        var line    : 2;
        var nind    : 2;
        var string  : 127;
        var nothing : 2;
        var ResourceIndex : 3;
        var hnombres : 2;
        var hfechas  : 2;
        var ctrvoz   :127;
        var resultado_rec:127;
enddec

program
        # The ResourceIndex to use is passed as an argument when you Run the Project
        # (See menu item Project/Configure Phonelines)
        # If no argument is passed, zero is used.
        # Note: Resources begin at 0.
        ResourceIndex = arg();
             line = ResourceIndex+1;

        # Reserve the resource indicated by 'ResourceIndex'.
        TrunkUse(ResourceIndex);

        ivr_recstr_release(line);
        ivr_playstr_release(line);

        ivr_unloadvcb(line, -1);

        # Wait for an inbound call.
        voslog("@L Waiting for incoming call");
        TrunkWaitCall();
```

```
# Answer the inbound call
TrunkAnswerCall();

# Check the results of the answer
if (TrunkGetState() strneq "Connected")
        # There was an error, so report the error and exit
        voslog("@E TrunkAnswerCall() failed. Check VOS1.LOG for details.");
        restart;
else
        # Process the inbound call
        if (ivr_getusedasrlic () > 0)
                ivr_recstr_wait(line, 1000);
        endif
        if (ivr_getusedttslic () > 0)
                ivr_playstr_wait(line, 1000);
        endif

        ivr_setparm(line, IVXCH_TTSSPKNAME, "carlos");
        ivr_playstr(line, "Bienvenido a la demostración del sistema IBERVOX de " &
                         "reconocimiento y conversión texto voz en castellano");
        while (1)

                hnombres = ivr_loadvcb(line, "nombres.txt", IVR_ISOLATED);

                # nombres.txt
                ivr_activatevcb(line, hnombres);
                ivr_playstr(line, "Tras oir la señal indique un nombre o departamento de la lista");
                ctrvoz = ivr_recstr(line, 1, 50, 35, 3);


                if ( ctrvoz eq 1 )
                    #se detecto tecla
                           resultado_rec=MediaWaitDigits(5,7);
                    resultado_rec=MediaGetDigitBuffer();
                endif

                ivr_playstr(line, "Gracias");

                nind = ivr_recind(line, 1);
                if (nind > 0)
                        if (ivr_score(line, 1) > 35)
                                string = "El nombre reconocido es " & ivr_word(line, ivr_id(line, 1));
                                nothing = 0;
                        else if (ivr_score(line, 1) > 25)
                                string = "El nombre más parecido de la lista es " & ivr_word(line, ivr_id(line, 1
                                nothing = 0;
                        else
                                string = "Lo siento, no se ha reconocido ningún nombre de la lista";
                                if (++nothing >= 10)
                                        break;
                                endif
                        endif
                        endif
                else
                        if (ivr_id(line, 1) eq IND_NOWORD)
                                string = "Lo siento, no se ha reconocido ningún nombre de la lista";
                        else
                                string = "Lo siento, no se ha reconocido nada";
                        endif
                        if (++nothing >= 10)
                                break;
                        endif
                endif
                ivr_playstr(line, string);
                ivr_deactivatevcb(line, hnombres);

                # fechas.bnf
                hfechas = ivr_loadvcb(line, "fechas.bnf", IVR_ABNF);
                ivr_activatevcb(line, hfechas);
                ivr_playrecstr(line, "Por favor, diga una fecha en cualquier momento.", 0, 50, 3);
                #ivr_recstrf(line, 1, 50, 35, 3, "RECF" & line & ".esu");
                ivr_playstr(line, "Gracias");

                nind = ivr_recind(line, 10);
                if (nind > 0)
                        if (ivr_recscore(line, "ddmm.dd") > 35 and ivr_recscore(line, "ddmm.mm") > 35)
                                string = "La fecha reconocida es " & ivr_recresult(line, "ddmm.dd") & " del " & i
                                nothing = 0;
                        else if (ivr_recscore(line, "ddmm.dd") > 25 and ivr_recscore(line, "ddmm.mm") > 25)
                                string = "La fecha más parecida es " & ivr_recresult(line, "ddmm.dd") & " del " &
                                nothing = 0;
                        else
                                string = "Lo siento, no se ha reconocido ninguna fecha";
                                if (++nothing >= 10)
                                        break;
                                endif
                        endif
                        endif
                else
                        if (ivr_id(line, 1) eq IND_NOWORD)
                                string = "Lo siento, no se ha reconocido ninguna fecha";
                        else
                                string = "Lo siento, no se ha reconocido nada";
                        endif
                        if (++nothing >= 10)
                                break;
                        endif
                endif
                ivr_playstr(line, string);
                ivr_deactivatevcb(line, hfechas);
```

```
                        ivr_setvcb(line, "builtin:grammar/number:es?max=1000", "0x00080000");
                        ivr_playrecaudio(line, "diganumero.esa", "0x0000002", 50, 3);
                        ivr_playstr(line, "Gracias");

                        nind = ivr_recind(line, 10);
                        if (nind > 0 and ivr_builtin(line) > 0)
                                if (ivr_recscore(line, "") > 35)
                                        string = "El número reconocido es " & ivr_recresult(line, "");
                                        nothing = 0;
                                else if (ivr_recscore(line, "") > 25)
                                        string = "El número más parecido es " & ivr_recresult(line, "");
                                        nothing = 0;
                                else
                                        string = "Lo siento, no se ha reconocido ningún número";
                                        if (++nothing >= 10)
                                                break;
                                        endif
                                endif
                                endif
                        else
                                if (ivr_id(line, 1) eq IND_NOWORD)
                                        string = "Lo siento, no se ha reconocido ningún número";
                                else
                                        string = "Lo siento, no se ha reconocido nada";
                                endif
                                if (++nothing >= 10)
                                        break;
                                endif
                        endif
                        ivr_playstr(line, string);
                        ivr_unloadvcb(line, -1);
                endwhile
        endif

        # Now that we are done, drop the call and restart
        TrunkDisconnect();
        restart;

endprogram

onhangup
        # The remote party hung up, so drop the call and restart.
        voslog("@L Caller hung up");
        TrunkDisconnect();
        restart;
end
```

2. Ejemplo

Ejemplo 4.2. Ejemplo de SpeakerID sobre CT ADE

```
# This program is required for every project.
# It is used to assign programs to trunks.
# Deleting or modifying it may cause your project to not run properly.
dec
        const IVR     = "0";
        const IBERVOX = "1";
        const PRODUCT = "0";

        const IVXGB_RECSTRLIC  = "201";
        const IVXGB_PLAYSTRLIC = "202";

        # VoxServer
        const IVXGB_DEFSERVER  = "801";

enddec
program
               # Funciones VERBIO
               ivr_setparm(0, IVXGB_DEFSERVER, "192.168.2.101");
               ivr_asr_init("es", "es");

               ivr_spk_init();

               vid_write("IVR RLL ", ivr_version(IVR, PRODUCT));
                vid_write("Verbio ", ivr_version(IBERVOX, PRODUCT));

                vid_write("ASR licenses (Spanish): ", ivr_getasrlic("es"));
                vid_write("TTS licenses (Spanish): ", ivr_getttslic("es"));

        Master("aeat");
endprogram

# Inbound - See ReadMe.Txt for information on running this sample
# and its requirements.

dec
        const IND_NOWORD       = "-1000";
        # IberVox Languages
        const LVX_SPANISH      = "0x00000000";
        const LVX_CATALAN      = "0x01000000";
        # IberVox Grammar Types
        const IVR_CONNECTED    = "0"; # connected words
        const IVR_ISOLATED     = "1"; # isolated words
        const IVR_ABNF         = "8"; # ABNF
        # Channel parameters
        const IVXCH_TTSSPEED   = "104";
        const IVXCH_TTSLNG     = "106";
        const IVXCH_TTSPITCH   = "107";
        const IVXCH_DEFASRLNG  = "10C";
        const IVXCH_TTSVOLUME  = "10D";
        const IVXCH_TTSSPKNAME = "402";

        var line    : 2;
        var nind    : 2;
        var string  : 127;
        var dtmf    : 10;
        var nothing : 2;
        var ret     : 16;
        var ResourceIndex : 3;
        var hlanguages : 2;
        var hdigits  : 2;
enddec

program
        # The ResourceIndex to use is passed as an argument when you Run the Project
        # (See menu item Project/Configure Phonelines)
        # If no argument is passed, zero is used.
        # Note: Resources begin at 0.
        ResourceIndex = arg();
             line = ResourceIndex+1;

        # Reserve the resource indicated by 'ResourceIndex'.
        TrunkUse(ResourceIndex);
        MediaUse(ResourceIndex);
        TopazRoute();

        # Wait for an inbound call.
        voslog("@L Waiting for incoming call");
        vid_write("Waiting for incoming call");
        TrunkWaitCall();

        # Answer the inbound call
        TrunkAnswerCall();

        # Check the results of the answer
        if (TrunkGetState() strneq "Connected")
                # There was an error, so report the error and exit
                voslog("@E TrunkAnswerCall() failed. Check VOS1.LOG for details.");
                restart;
        else
                # Process the inbound call
```

```
            if (ivr_spk_exist(line, "1234") eq 0)

                    ret = ivr_spk_addfile(line, "1234", "C:/CT ADE Projects/SpeakerID/Audios/77308296.wav", "77308296");
            vid_write("spk_add_file returned: ", ret);
                    ret = ivr_spk_addfile(line, "1234", "C:/CT ADE Projects/SpeakerID/Audios/0123456789.wav", "0123456789");
                    vid_write("spk_add_file returned: ", ret);
                    ret = ivr_spk_addfile(line, "1234", "C:/CT ADE Projects/SpeakerID/Audios/9876543210.wav", "9876543210");
                    vid_write("spk_add_file returned: ", ret);
                    ret = ivr_spk_addfile(line, "1234", "C:/CT ADE Projects/SpeakerID/Audios/0246813579.wav", "0246813579");
                    vid_write("spk_add_file returned: ", ret);
                    ret = ivr_spk_addfile(line, "1234", "C:/CT ADE Projects/SpeakerID/Audios/1470258369.wav", "1470258369");
                    vid_write("spk_add_file returned: ", ret);
                    ret = ivr_spk_addfile(line, "1234", "C:/CT ADE Projects/SpeakerID/Audios/5193280476.wav", "5193280476");
                    vid_write("spk_add_file returned: ", ret);
                    ret = ivr_spk_addfile(line, "1234", "C:/CT ADE Projects/SpeakerID/Audios/9461725038.wav", "9461725038");
                    vid_write("spk_add_file returned: ", ret);
                    ret = ivr_spk_addfile(line, "1234", "C:/CT ADE Projects/SpeakerID/Audios/5862319407.wav", "5862319407");
                    vid_write("spk_add_file returned: ", ret);
                    ret = ivr_spk_addfile(line, "1234", "C:/CT ADE Projects/SpeakerID/Audios/3659102487.wav", "3659102487");
                    vid_write("spk_add_file returned: ", ret);
                    ret = ivr_spk_addfile(line, "1234", "C:/CT ADE Projects/SpeakerID/Audios/4209167183.wav", "4209167183");
                    vid_write("spk_add_file returned: ", ret);

                    ret = ivr_spk_train(line, "1234", "");
                    vid_write("ivr_spk_train returned: ", ret);
            else
                    ret = ivr_spk_verify(line, "1234", "C:/CT ADE Projects/SpeakerID/Audios/verify.wav", "02468");
                    vid_write("ivr_spk_verify returned: ", ret);
            endif


    endif

    # Now that we are done, drop the call and restart
    TrunkDisconnect();
    restart;

endprogram

onhangup
    # The remote party hung up, so drop the call and restart.
    voslog("@L Caller hung up");
    TrunkDisconnect();
    restart;
end
```

3. Apéndice A

nombres.txt

```
Eduardo Campillo
Asunción Chinea
Marta Cuenca
Emilio Doblado
Guillermo Estrada
María Frías
Sergio Galíndez
Eloy de la Fuente
Enrique Hurtado
Pablo Jiménez
Francisco Luna
Fernando Mayoral
Susana Martínez
Germán Morales
Alfonso Morillo
Juan Oliva
Inmaculada Sánchez
Gloria Machado
Pilar Zaragoza
Irene Ponsa
Jose Riveiro
Marcial Rodríguez
Jorge Sánchez
Alberto Salvador
Héctor Osorio
Carmen Puente
Carolina Segovia
Alejandro Tarrida
Leonardo Toronto
Jesús Urrubieta
Departamento Comercial
Departamento Técnico
Proveedores
Administración
Dirección
```

fechas.bnf

```
#ABNF 1.0 ISO8859-1;

/****************************************************************
 $x = y    : rule
 $x        : local reference
 x, "x y"  : tokens
 x y       : sequence of tokens
 x | y     : alternatives
 [x]       : x is optional
 x *       : x repeated zero or more times
 x +       : x repeated one or more times
 x {y}     : when x is recognized, system returns y
 (...)     : grouping

 public $x = y;   : public rule.
 root $x = y;     : main public rule.

 ASR uses root rule. If root rule not declared, main rule is
 composed of all public rules ($x | $y | ...)

 language es; : sets default language (es | ca)
 x!ca         : word x in language ca
****************************************************************/

/****************************************************************
 Example 1:
 ----------
 $rgb  = rojo {R} |
         verde {G} |
         azul {B} |
         "azul marino" {B}
         blau!ca {B};

 $fill = color | "el color" | el;

 root $color = [SIL] [$fill] $rgb ["por favor"] [SIL];

 Example 2:
 ----------
 public $dayweek = lunes | martes | miércoles | jueves |
                   viernes | sábado | domingo;
 public $dayrel  = hoy | mañana | "pasado mañana" | ayer;

 (implicit declaration of "root $day = $dayweek | $dayrel;")

****************************************************************/

// day

$dd = uno {1} | dos {2} | tres {3} | cuatro {4} | cinco {5} |
      seis {6} | siete {7} | ocho {8} | nueve {9} | diez {10} |
```

```
      once {11} | doce {12} | trece {13} | catorce {14} | quince {15} |
      dieciséis {16} | diecisiete {17} | dieciocho {18} | diecinueve {19} |
      veinte {20} | veintiuno {21} | veintidós {22} | veintitrés {23} |
      veinticuatro {24} | veinticinco {25} | veintiséis {26} | veintisiete {27} |
      veintiocho {28} | veintinueve {29} | treinta {30} | treinta_y_uno {31};

// month

$mm = enero {1} | febrero {2} | marzo {3} | abril {4} | mayo {5} |
      junio {6} | julio {7} | agosto {8} | septiembre {9} | octubre {10} |
      noviembre {11} | diciembre {12} |

      uno {1} | dos {2} | tres {3} | cuatro {4} | cinco {5} | seis {6} |
      siete {7} | ocho {8} | nueve {9} | diez {10} | once {11} | doce {12};


// silence: model [SIL] or [H] (very short silence)

$SIL = "[SIL]" {} | "[H]" {};

// uno de enero
// el día [H] dos de febrero
// dia cinco
// el siete
// el día siete [SIL] del tres

root $ddmm = $SIL* [el | el_día] $dd [$SIL* (de | del) $mm] $SIL*;
```

Índice